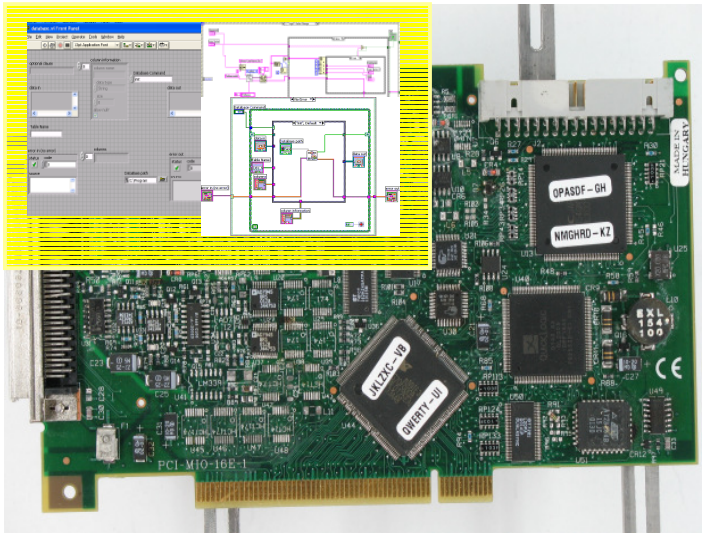
VPIS

Visual Product Inspection System



Vizuális Termékellenőrző Rendszer

Adatbáziskezelése, adatbázis rendszerének fejlesztése

TARTALOMJEGYZÉK

BEVEZETÉS

Az 1940-es évek végén egy új technológiai vívmány került az emberiség szolgálatába. Az elektronika és az elektromosság kutatásával párhuzamosan, a felmerülő igények mellett, rengeteg energiát fektettek a gépek tervezésébe és kivitelezésébe. Főleg olyan területeken próbálták az emberi munkát felváltani, ahol az lehetetlennek, veszélyesnek vagy körülményesnek bizonyult, mint például sugárveszélyes helyeken.

A kezdeti gépépítők fantáziája azonban nem állt meg. A praktikus és a gyakorlatban használható gépek mellett a kutatások abba az irányba fordultak, hogy az emberi munka és erő mellett az irányítás, egyszerű döntési mechanizmusok, minőségellenőrző feladatok megoldására is képesek legyenek a gépek.

Manapság a gyártási folyamatok majdnem teljes egészét gépek végzik, az emberek csupán felügyelik a gépek helyes működését. A gyártás során elkészült termékeket szintén gépek ellenőrzik, hogy minden alkatrész a helyére került-e, tesztelik, hogy helyesen működnek-e a piacra kerülés előtt.

Egy elektronikus készülék felépítésének egyes szakaszaiban egymástól alapvetően eltérő vizsgálati technikákat használnak. A tokozatlan integrált áramkörökhöz rugalmas tűkön keresztül lehet csatlakozni, a vizsgálatot erre a célra kifejlesztett úgynevezett szelet-mérő automata végzi. A tokozott áramkör alkatrész-adagolóból kerül a vizsgálati befogóba, itt egy alkatrész-vizsgáló automata vizsgálja a működés helyességét.

A panelbe szerelt alkatrész jóságát tűágyon, többnyire in-circuit mérőautomatával vizsgálják, a panel egészének működését pedig funkcionális tesztelővel ellenőrzik. Az összeszerelt részegység illetve készülék vizsgálathoz egyedi műszer-összeállítású és programozású mérőautomata szükséges.

A gyártási folyamat során többször végeznek optikai vizsgálatot a paneleken. Ezek a beültetett alkatrészek meglétére, pontos elhelyezésére, polaritás vizsgálatára vonatkoznak. A gyakorlatban ezt az ellenőrzést többnyire emberi munkával végzik. Az emberi szem nem mindig megbízható, fáradékony, ezért gondoltunk a feladat automatizálására.

A TÉMAVÁLASZTÁS INDOKLÁSA

Programozó matematikus szakon megfogott a LabView grafikus programozói felülete. Nem csupán kódolásból áll egy program megírása, hanem „kézzel fogható” elemek összekapcsolásával lehet műveleteket végrehajtani. A már meglévő elemeket egymáshoz kapcsolhatjuk, kedvünk szerint pakolhatjuk, melyből saját VI elemeket készíthetünk. Ez a tevékenység a gyermekkori építőjátékokra emlékeztet. De az emberek felnőtt korukban is szeretnek játszani…

A LabVIEW programokat virtuális műszereknek, vagy rövidebben VI – oknak nevezzük, mert megjelenésükben és működésükben a fizikai műszereket utánozzák, de igen széles eszközkészlettel rendelkeznek az adatgyűjtés, adatelemzés, megjelenítés és adattárolás területén.

Ma már mérnökök és kutatók a világ minden táján számtalan iparágban használják ezt a programozási nyelvet. Autók, telekommunikációs eszközök, félvezetők tervezéséhez és gyártásához, biológiai és kémiai kutatásokhoz stb. Kiegészítő modulok és eszközkészletek használatával olyan speciális függvényekkel és funkciókkal bővíthetjük ki a LabVIEW-t, amelyek különböző felhasználási területeken nyújtanak gyors és hatékony megoldást.

Egyetemi tanulmányaim során a LabView alapvető tulajdonságait sajátítottuk el, melyet később képfeldolgozási ismeretekkel bővítettünk. Ezzel kapcsolatos érdeklődésből alakult ki a VPIS – Vizuális Termékellenőrző - rendszer, amelyet három nagy témára bontottuk. Ezek a kamerára vonatkozó ismeretek, a vizsgálatot végző képfeldolgozás és az eredmények eltárolására, statisztikai adatok nyerésére vonatkozó adatbáziskezelés.

Az én feladatom a csapatmunkában az adatbáziskezelés, amely a programozás elengedhetetlen része, így itt is végigköveti a rendszer teljes folyamatát.

Adatbázis-kezelésre vonatkozó ismeretekkel rendelkezem, de a LabView adatbáziskezelő elemeivel korábban még nem találkoztam, éppen ezért dolgozatom elején szeretném bemutatni a Database Connectivity Toolkit elemeit, majd ezen elemek felhasználási lehetőségeit. Dolgozatom második felében rátérek a VPIS rendszer adatbázis-kezelésére vonatkozó igényekre, bemutatva az eddig elkészült és a rendszerbe beépített elemeket.

A TÉMA FELDOLGOZÁSÁHOZ SZÜKSÉGES RENDSZERKÖVETELMÉNY

A szakdolgozat elkészítéséhez rendelkezésemre álló számítógép hardver jellemzői:

Processzor: Mobile AMD Athlon™ XP 1800+ 1,53 GHz

Memória: 256 MB RAM

Winchester: 20 GB

Optikai meghajtó: combo

A szakdolgozat elkészítéséhez rendelkezésemre álló számítógép szoftver jellemzői:

Microsoft Windows XP Professional 2002-es verzió

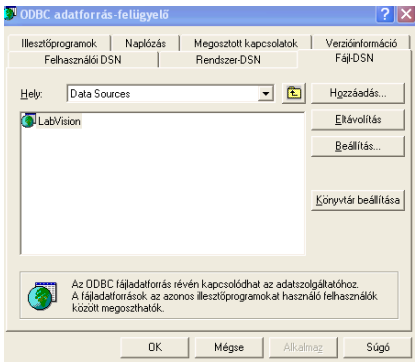
Microsoft Office Access 2003

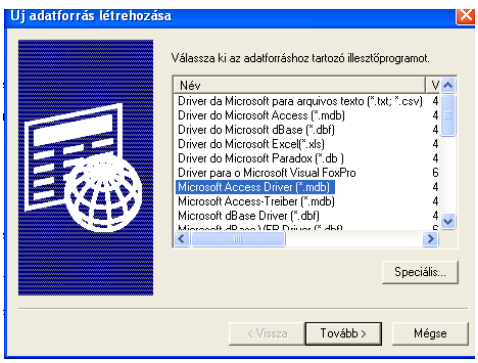
LabVIEW 7.1 , LabVIEW 8.0, Vision Assistant 7.1, Vision Assistant 8.0

Telepített Database Connectivity Toolkit (LabVIEW 7.1)

A szakdolgozat működéséhez szükséges egyéb rendszerbeállítások:

A VPIS adatbázis használata miatt a rendszer működéséhez szükség van az ODBC adatforrás megfelelő beállítására, amit a „Felügyeleti eszközök” rész alatt található „ODBC adatforrás”-nál tehetjük meg a megfelelő Fájl-DSN megadásával. A hozzáadás gombra kattintva kiválasztjuk a megfelelő adatbázis drivert, ami a VPIS rendszernél a Microsoft Access Driver, majd meghatározzuk az adatforrás nevét és helyét, amelyet a Vison rendszerünk használ.





NI – NATIONAL INSTRUMENTS

A National Instruments™ 1976-ban, három mérnök kezdeményezésére jött létre a texasi Austinban. Már a kezdeti kutatások célja is az volt, hogy a mikroelektronika eredményeit a korszerű mérnöki tudással kombinálják; a kutatás és fejlesztés a mai napig a National Instruments™ fő tevékenysége maradt. A kezdetek óta eltelt több évtized során a vállalat piacvezető pozíciót szerzett a mérés- és automatizálás technológia terén, s számos iparágban megvetette lábát állandó és folyamatosan bővülő ügyfélkörének megteremtésével.

A NI Corporation ma több mint 30 országban rendelkezik értékesítési irodával. Míg hardver-termelés csak Austinban és Debrecenben folyik, kutatás-fejlesztési központok az Egyesült Államokban, Németországban és Kínában vannak. A társaság dolgozóinak összlétszáma meghaladja a 3500 főt.

A hajdani alapítók közül Dr. James Truchard a National Instruments™ jelenlegi elnök vezérigazgatója, s egyben a vállalati kultúra egyik meghatározója is. Utóbbira jellemző, hogy az egykor néhány fős vállalkozásból a bővülés és multinacionális vállalattá válása után sem veszett ki a munkatársak közötti közvetlenség. A vezetőség tudatosan törekszik a felszabadult légkör megteremtésére szervezeti egységeiben, hiszen a vállalat sikerének kulcsául a dolgozók motiváltságát és elégedettségét tekinti. Ez a típusú gondolkodásmód természetesen a debreceni gyárra is érvényes, a több fős HR (Human Resources – Emberi Erőforrás) csapat a vezetőséggel együttműködve gondoskodik a jelenleg több mint 300 munkatárs szakmai fejlődéséről és a jó csapatszellemről

A National Instruments™ Europe Kft. 2001-ben nyitotta meg kapuit és indította el gyártósorait Kelet-Magyarország egyik legnagyobb volumenű zöldmezős beruházásaként a debreceni Ipari Parkban. Az ünnepélyes alapkőletétel 2001. májusában volt, és szeptember végén már legördült az első kártya az első sorról… [4]

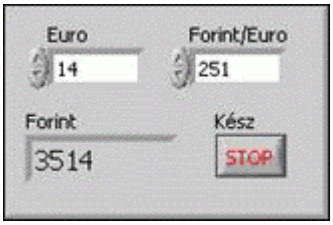
LABVIEW

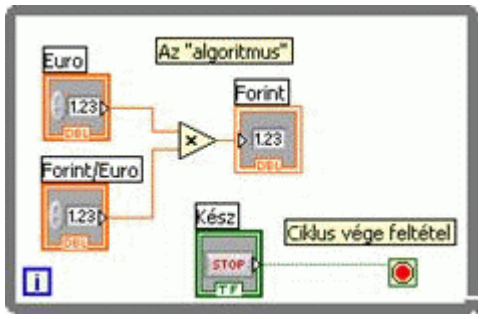
A G-nyelv, a Labview, a National Instruments által kifejlesztett vizuális programozási nyelv. A nyelvet méréstechnikai célokra (adatgyűjtés, jelfeldolgozás, folyamatirányítás) alkották, de általános célra is jól használható, grafikus adatfolyam nyelv.

A programírás a modulok közti adatkapcsolatok vezetékezéséből áll. Minden program vagy szubrutin - VI , virtuális eszköz - két részből áll, előlapból és hátlapból, blokkdiagramból.

Az előlap a felhasználói felület, beavatkozó szervekkel és kijelzőkkel. A hátlap a kód, az adatáramlási diagramm. [5]

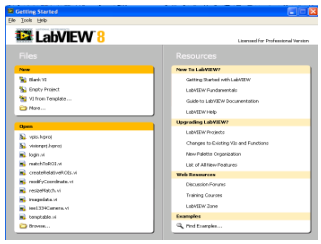
Az alábbi "program" a stop gomb lenyomásáig konvertál a bevitt pénzösszeg és árfolyam alapján





A LabVIEW 1986-os megjelenését követően rohamléptekben hódította meg a mérnökvilágot. A mérnökök és kutatók ma már a LabVIEW-t az elsőszámú fejlesztőeszközként tartják számon mérő-, adatgyűjtő- és automatizálási feladatok megoldásához.

A National Instruments által kifejlesztett LabVIEW sikere a sokoldalúságában, egyszerűségében és hatékonyságában rejlik. A PC alapú adatgyűjtő és automatizálási feladatok megoldásához kifejlesztett LabVIEW egy olyan grafikus fejlesztőkörnyezet nyújt, amely segítségével egyszerű és komplex, nagy megbízhatóságú alkalmazásokat is gyorsan elkészíthetünk.



Teljeskörű támogatás

A LabVIEW gyors elterjedését többek között a támogatott eszközök széles skálája tette lehetővé. A National Instruments teljes termékkínálatán felül már több mint 150 gyártó 2200 termékét képes kezelni, ezáltal egy közös szoftverplatformot nyújtva az eszközök integrálásához. A LabVIEW támogatja többek között a soros, GPIB, PXI, CAN, USB, Ethernet és wireless kommunikációt. Egy szabványos ún. VISA (Virtual Instrumentation Software Architecture) eszközvezérléssel lehetővé teszi, hogy az elkészített alkalmazást - nevezzük virtuális műszernek (**V**irtual **I**nstruments) - függetlenítsük az alkalmazott eszköz kommunikációs szabványától. Ezáltal, a virtuális műszer módosítása nélkül, könnyedén felcserélhetünk például egy soros kommunikációt támogató eszközt egy GPIB kommunikációt használó eszközre. A virtuális műszerek egyszerű hordozhatósága egyaránt biztosított, mivel LabVIEW-val egyszerűen készíthetünk alkalmazásainkból EXE és DLL fájlokat.

Grafikus, könnyen kezelhető felület

A siker kulcsa a szoftver grafikus fejlesztőfelületében rejlik. A szöveges programozási nyelvek összes funkcionalitásának megtartása mellett nyújt magas szintű grafikus fejlesztőeszközt mérési és automatizálási feladatok gyors megoldásához. A blokkokból felépülő LabVIEW alkalmazás egy olyan virtuális eszköz, amely egy blokkdiagrammot és egy hozzá szorosan kapcsolódó vezérlőpanelt tartalmaz. A blokkokból felépített kód jól átlátható és később szükség esetén könnyen módosítható. Saját virtuális eszközünket a több száz előre definiált függvényblokk és az új Express technológia segítségével pillanatok alatt elkészíthetjük. Egyszerűbb mérő és adatgyűjtő alkalmazások percek alatt felépíthetőek, de bonyolult vezérlő és tesztrendszerek alkalmazásfejlesztési idejét is töredékére csökkenthetjük. A DAQ Assistant segítségével egyszerűen definiálhatjuk a mérési feladatnak megfelelő analóg és digitális I/O csatornákat. A .NET és ActiveX szoftver technológiák támogatásával lehetővé válik más Windows alkalmazások használata is. Rövid idő alatt például olyan WEB-es alkalmazást készíthetünk, amely lehetővé teszi a méréseink távoli számítógépről való vezérlését, monitorozást és előre definiált események esetén az e-mail küldését.

Mivel a LabVIEW az egyetlen grafikus fejlesztőkörnyezet, amely a C programokhoz hasonló futási sebességű kód generálására alkalmas kódoptimalizáló fordítóval rendelkezik, ezért futásidő kritikus rendszerek gyors megvalósítására is a legalkalmasabb fejlesztőeszköz.

Sokoldalúság

A LabVIEW-t ma már mérnökök és kutatók a világ minden táján számtalan iparágban használják, például autók, telekommunikációs eszközök, félvezetők tervezéséhez és gyártásához, biológiai és kémiai kutatásokhoz stb. Kiegészítő modulok és eszközkészletek használatával olyan speciális függvényekkel és funkciókkal bővíthetjük ki a LabVIEW-t, amelyek különböző felhasználási területeken nyújtanak gyors és hatékony megoldást.

Kiegészítő eszközkészletek:

- Adatbázis kapcsolat
- Riport generálás a Microsoft Office-hoz
- Jelfeldolgozás
- Digitális szűrő tervezés
- Hang és rezgés analízis
- PID szabályozás stb.

A kiegészítő modulok:

- Real-Time modul
- FPGA modul
- Vision Development modul stb.

A Real-Time modul segítségével olyan valós idejű alkalmazást készíthetünk, ami a célhardverre töltve determinisztikusműködést biztosít. A célhardver lehet egy valós idejű operációs rendszert futtató NI eszköz (pl. PXI, FieldPoint, CompactVision, CompactRIO), vagy egy személyi számítógép. A Real-Time modul a robosztus, kifejezetten terepi mérésadatgyűjtéshez kifejlesztett valós idejű futást biztosító FieldPoint rendszerrel tökéletes megoldás elavult PLC-k lecseréléséhez.
Az FPGA modul egy olyan könnyen kezelhető grafikus fejlesztőeszközt nyújt az újraprogramozható FPGA-val rendelkező CompactRIO-hoz, amely egyedi mérő és vezérlő eszközök készítését teszi lehetővé. [6]

A Vision Development modul az ipar számos területén (pl. minőségellenőrzés) alkalmazott gépi látáshoz és speciális képfeldolgozási alkalmazások fejlesztéséhez kínál teljeskörű megoldást. A National Instruments a LabVIEW mellett olyan teljes értékű fejlesztőeszközöket is kínál, amelyek a LabVIEW-val együttműködve képesek összetett tesztrendszerek megvalósítására (TestStand), komplex matematikai és vizuális adatelemzés elvégezésére akár 100 billió adattal (DIAdem) vagy a prototípus gyors megtervezésére és tesztelésére (Signal Express).

Tökéletes választás

Ma már a világon mérnökök és kutatók ezrei használják a LabVIEW-t. A felhasználók elismerését számos nemzetközi díj is fémjelzi. A LabVIEW tökéletes választást jelent, ha egy olyan könnyen kezelhető és sokoldalú szoftvert szeretnénk kapni, amely gyors és hatékony megoldást nyújt a mérés és automatizálás minden területén.

A National Instruments régóta használja saját termékét a mérés és irányítástechnikában, az automatizálásban és egyéb területeken, de a képfeldolgozási műveletek végzésére még nem alkalmazták. Vállalkozó egyetemistákként ezen a téren próbáltunk elindulni, módszereket keresni az alkalmazására a cég segítségével.

A VPIS ADATBÁZISRENDSZERE

A megfelelő adatbázis rendszer kiválasztása nem volt egyszerű dolog. A LabView adatbázis elemei a Microsoft Access-re készültek, de más adatbáziskezelő rendszereket is támogat. A Campus Licence keretében jogtiszta Microsoft Office termékcsaláddal rendelkeznek a felsőoktatás hallgatói és oktatói, ezért választottam az Access 2003-at a VPIS adatbázisaként.

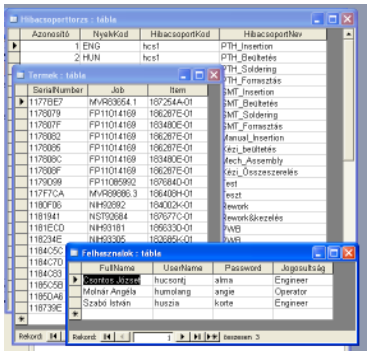
Microsof Office Access 2003

Az Access relációs adatbázisokat hoz létre. Ez azt jelenti, hogy a program az adatokat tárgy vagy feladat szerint különböző, különálló táblákban tárolja, de ezek az adatok összefüggenek, és a megadott módon összeegyeztethetők. Bármikor képes összegyűjteni a felhasználó számára szükséges információkat, azok könnyedén kinyomtathatóak. Két adatkészlet egymáshoz képest relatív, az egyik adatkészletben lévő információk kapcsolatban vannak a másik listában lévő megfelelő adatokkal, „ismerik" azokat.
Az adatbázis legjobb kihasználása érdekében az adattáblákat érdemes úgy beállítani, hogy azok tükrözzék az adatokhoz tartozó tárgyakat, feladatokat.
Az adatbázis tervezésekor érdemes figyelembe venni, a felhasználók igényeit az adatok kikeresésére, jelentés készítésére vonatkozóan. Egy kevés előrelátás igen hasznos lehet.

Az Access adatbázisok objektumokból állnak. A négy legfontosabb ezek közül:

Táblák:
Sorokban és oszlopokban tárolják az adatokat. Minden adatbázis egy vagy több táblát tartalmaz.
Lekérdezések:
Az adatok lekérdezésére és feldolgozására szolgálnak. Egyesítik a különböző táblákban lévő adatokat, frissítik azokat, és számításokat végeznek az adatokon.
Űrlapok:
Az adatbevitel és az adatnézetek vezérlésére szolgálnak. Az adatokat vizuálisan jelenítik meg, így könnyebb a munka.
Jelentések:
Az adatok összesítésére és kinyomtatására szolgálnak. A táblákban és lekérdezésekben szereplő adatokat dokumentumokká formálják az ötletcsere érdekében.

Az Access alkalmazásban az összes adatbázis legalább egy táblát tartalmaz. A táblák tartalmazzák az adatokat, így azok bármely adatbázis nélkülözhetetlen építőelemei.

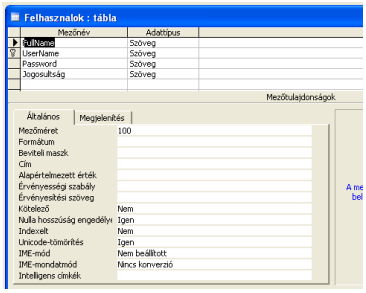


Az adatbázisnak külön táblában kell tárolnia minden jelentősebb témakört. Például a termékek információit, a felhasználók adatait, vagy a hibajelzésekre vonatkozó adatokat. Az adatismétlés gyakori hiba, de a táblák megfelelő strukturálásával könnyedén elkerülhető.

Minden tábla *rekord*oknak nevezett sorokból és *mező*knek nevezett oszlopokból áll. A Felhasználók tábla egyik *rekord*ja például egy személyre vonatkozó összes adat, amit az adott tábla tartalmaz. A *mező* pedig vonatkozhat például a Felhasználók nevére.

Az adatbázis mezőinek *beállítás*ai határozzák meg a tárolható adatok típusát, az adatok megjelenését és azt, hogy az adatok mire használhatóak.

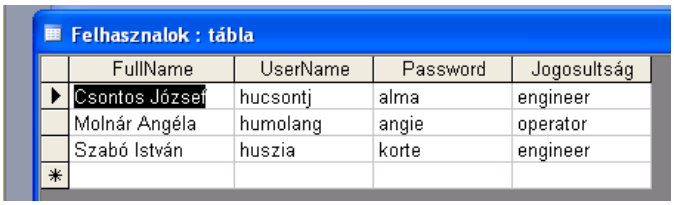
A mezők egyik legfontosabb beállítása az *adattípus*, amely szöveg, szám, pénznem (pénz), logikai, bináris valamint dátum/idő lehet. Az adattípus korlátozza és leírja, hogy a mezőben milyen típusú adatok szerepelhetnek. Az adattípus meghatározza a mezőn végrehajtható műveleteket is, valamint azt, hogy az adat mennyi memóriát használ fel. A mezőknek vannak tulajdonságai is, amelyek a mezőn belül lévő információk részleteit ellenőrzik: a hosszúságot (karakterszámban megadva), az alapértéket, illetve valamely érvényességi szabályt, ami biztosítja, hogy az adat megfeleljen bizonyos feltételeknek. A tulajdonságok egyszerűbbé teszik az adatok beírását és kezelését.



Az egyes táblákban minden egyes rekordnak egyedinek kell lennie. Annak érdekében, hogy két rekord megkülönböztethető legyen egymástól, a táblák *elsődleges kulcs* mezőt tartalmaznak. Ez egy olyan azonosító, termékkód, egyedi kód, amely minden egyes rekordnál egyedi. Ha szükséges, az Access programban autoincrement numerikus elsődleges kulcsot rendelhetünk a rekordokhoz. Ez a kulcs mindig egyel növekszik, amikor rekordot veszünk fel a táblába. Az elsődleges kulcs megkülönbözteti a hasonló adatokat, és minden egyes rekordot egyedivé tesz. Segítségével össze is kapcsolhatjuk az adatokat. Egy táblát elsődleges kulcs segítségével kapcsolhatunk egy másik táblához. Ebben a relációban az egyik tábla elsődleges kulcsa a másik tábla idegen kulcsává válik. [7]

A VPIS adattáblái:

Felhasználók:

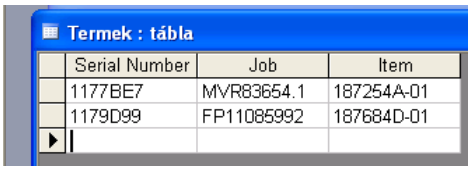

Felhasznalok : tábla

FullName	UserName	Password	Jogosultság
Csontos József	hucsontj	alma	engineer
Molnár Angéla	humolang	angie	operator
Szabó István	huszia	korte	engineer

A szoftvert csak arra jogosult személyek használhatják. Az adatbázis egyik fontos táblája tartalmazza azoknak a személyeknek a legfontosabb adatait, akik beléphetnek a rendszerbe, és jogosultságuktól függően hajthatnak végre műveleteket.

A tábla tartalmazza a felhasználók teljes/hivatalos nevét, egy maximum 8 karakter hosszú felhasználói nevet a hozzá tartozó jelszóval és szerepel a személy jogköre. Ez két fajta lehet, operátor vagy mérnök. Az operátor csak termékellenőrzést végezhet a rendszerrel. A mérnök jogköre ennél sokkal bővebb. Ő a szoftver bármely funkcióját használhatja. Új felhasználót vihet a rendszerbe, statisztikai adatokat kérdezhet le, vizsgálati paraméterek alapján újabb ellenőrzésekre taníthatja be a rendszert.

Termékek:


Termek : tábla

Serial Number	Job	Item
1177BE7	MVR83654.1	187254A-01
1179D99	FP11085992	187684D-01

A gyártott termékek három fő tulajdonsággal rendelkeznek. Az egyik a Job szám, amely azt határozza meg, hogy mely termékcsaládról van szó. Egy új termék gyártásának bevezetésekor meghatározzák ezt a számot. Ezen belül vagy egy Item szám, amely a termékfajtát, a termékcsaládon belüli csoportot azonosítja. A Serial Number egy egyedi azonosító. Az azonos Job és Item számú termékeket ez alapján különböztetik meg. A szériaszám meghatározza a hozzá tartozó Job és Item számot is.

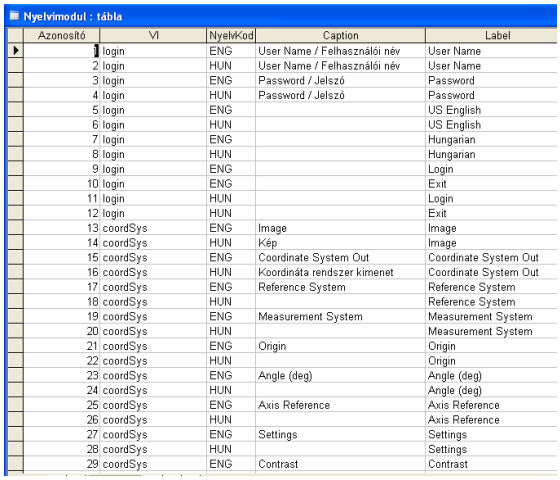
Nyelvtörzs


Nyelvtorzs : tábla

NyelvKod	Nyelv
HUN	Magyar
ENG	English

A szoftver magyar és amerikai felhasználók számára készült elsősorban, de a továbbiakban más nyelve való bővítésre is lehetőséget nyújtva egy külön táblában szerepelnek az erre vonatkozó legfontosabb adatok. Tároljuk a nyelv megnevezését és egy hozzá tartozó egységesen három karakteres kódot.

Nyelvi modul:

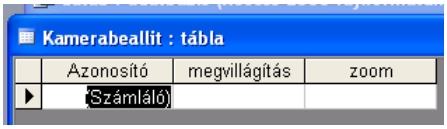

Nyelvimodul : tábla

Azonosító	VI	NyelvKod	Caption	Label
1	login	ENG	User Name / Felhasználói név	User Name
2	login	HUN	User Name / Felhasználói név	User Name
3	login	ENG	Password / Jelszó	Password
4	login	HUN	Password / Jelszó	Password
5	login	ENG		US English
6	login	HUN		US English
7	login	ENG		Hungarian
8	login	HUN		Hungarian
9	login	ENG		Login
10	login	ENG		Exit
11	login	HUN		Login
12	login	HUN		Exit
13	coordSys	ENG	Image	Image
14	coordSys	HUN	Kép	Image
15	coordSys	ENG	Coordinate System Out	Coordinate System Out
16	coordSys	HUN	Koordináta rendszer kimenet	Coordinate System Out
17	coordSys	ENG	Reference System	Reference System
18	coordSys	HUN		Reference System
19	coordSys	ENG	Measurement System	Measurement System
20	coordSys	HUN		Measurement System
21	coordSys	ENG	Origin	Origin
22	coordSys	HUN		Origin
23	coordSys	ENG	Angle (deg)	Angle (deg)
24	coordSys	HUN		Angle (deg)
25	coordSys	ENG	Axis Reference	Axis Reference
26	coordSys	HUN		Axis Reference
27	coordSys	ENG	Settings	Settings
28	coordSys	HUN		Settings
29	coordSys	ENG	Contrast	Contrast

A nyelvi modul tábla tartalmazza a VIokhoz tartozó elemek megjelenítendő neveit. A táblában megtaláljuk, a VI nevét, az elem Caption tulajdonságát a programozás során milyen névvel láttuk el, a nyelvkódot, és hogy az adott nyelvkód alapján a Caption tulajdonsághoz tartozó Label megnevezést.

Kamerabeállítások:

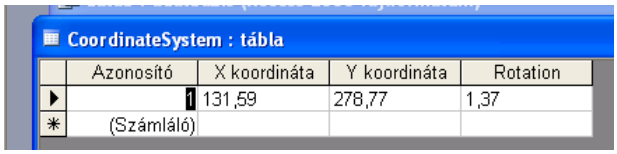
A vizsgálandó termékről először egy képet kell készíteni. Ez történhet ipari kamerával, vagy digitális fényképezőgéppel. Mindkét esetben a készülék össze van kötve a számítógéppel, az elkészült kép betöltődik a programba és kezdődhet a vizsgálat, a képfeldolgozás. A kamera beállításához szükséges adatok ebben a táblában tárolódnak. Tartalmazza a megvilágítás fokát és a képkészítési távolságot.


Kamerabeallit : tábla

Azonosító	megvillágítás	zoom
(Számláló)		

CoordinateSystem

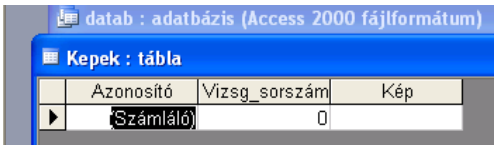
A képfelismeréshez szükséges koordináta rendszer adatokat tároljuk ebben a táblában. A panelről készült kép beolvasása után az első feladat a képre illeszthető koordináta rendszer meghatározása. Ennek X és Y koordinátái és az elfordulás szögét meghatározó értékek kerülnek tárolásra ebben a táblában.


CoordinateSystem : tábla

Azonosító	X koordináta	Y koordináta	Rotation
1	131,59	278,77	1,37
(Számláló)			

Képek

A vizsgálat során készült képeket eltároljuk a számítógépen, hogy bármikor visszakereshető legyen a vizsgálat eredményével együtt. Az adatbázisban a képekre csak hivatkozunk. A tábla így csak egy egyedi azonosítót tartalmaz, egy vizsgálati sorszámot, amely a képhez hozzákapcsolja a vizsgálati eredményeket egy másik táblából és a kép hivatkozását, hogy a számítógépen hová mentetettük el az adott vizsgálat során készült képet.


datab : adatbázis (Access 2000 fájlformátum)

Kepek : tábla

Azonosító	Vizsg_sorszám	Kép
(Számláló)	0	

Vizsgálatok

Egy vizsgálat elvégzése során keletkezett adatokat a Vizsgalat táblában tároljuk el. Ez tartalmaz általános adatokat, mint a vizsgált dátuma, mely operátor végezte a vizsgálatot, milyen felhasználói névvel jelentkezett be. A termékre vonatkozó tulajdonságokat, mint Job, Item és szériaszám. A vizsgálathoz tartozó kamera-beállítási paramétereket. A vizsgálat során készült két dokumentációra is hivatkozunk itt. Egyik a panelről a vizsgálat során készített kép, a másik a képfeldolgozás eredménye, a vizsgálat folyamán kimutatott hibák listája.

Vizsgalat : tábla

	Azonosító	SerialNumber	Job	Item	operator	kép	Hibalista	Datum	Kamerabeallitas
▶	(Számláló)					0	0		0

Hiba lista próba:

Egy panel ellenőrzése során egyetlen – a panelről készült - kép alapján nagyon sok információhoz juthatunk. Többféle vizsgálatot végezhetünk, amik eredménye során előfordulhatnak eltérések. Ezeket a hibákat egy külön táblában tárolja az adatbázis. A hibáknak van egy meghatározott kódolási rendszere, a szoftver is ezt az NI központi hiba-kódrendszert használja. Tárolásra kerül a hiba helye a pozíció alapján meghatározva, és a hiba pontosítása érdekében a tábla tartalmaz egy megjegyzés nevű oszlopot is. Ide bármi írható az operátor által.

A redundancia elkerülése érdekében a szériaszám azonosítja a hibákat ebben a táblában, hogy melyik panel vizsgálata során keletkezett az adott bejegyzés. Az Item számra a statisztika miatt van szükség, hiszen a széria szám egyedileg azonosítja a panelt, vele a konkrét hibát konkrét panelhez tudjuk párosítani.

HibaListaProba : tábla

	Azonosító	Item	SerialNumber	Megjegyzés	Pozicio	HibaKod	OpDontes
▶	+	1	187254A-01	1177BE7	C8	3	470 javitas
	+	2	187254A-01	1177BE7	C14	4	470 javitas
	+	3	187254A-01	1177BE7	U30	8	470 javitas
	+	4	187254A-01	1177BE7	U44	15	463 selejt
	+	5	186287E-01	1178079	J22	4	0 javitas
	+	6	186287E-01	1178079	J1	20	517 javitas
	+	7	186287E-01	1178079	C23	5	470 javitas
	+	8	186287E-01	1178079	R45	4	540 javitas
	+	9	186287E-01	1178079	U22	22	517 javitas
	+	10	186287E-01	1178082			0 rendben
	+	11	186287E-01	1178085	U22	22	517 javitas
	+	12	186287E-01	117807F	R45	4	540 javitas
	+	13	186287E-01	117808C			0 rendben
	+	14	186287E-01	117807F	U21	6	612 javitas
*		(Számláló)					0

A statisztika során arra vagyunk kíváncsiak, hogy egy fajta termék, amelyből meghatározott darabszámot gyártanak, milyen típusú hibákkal gyártódik le, mekkora százalékban fordulnak elő hibás termékek. Ezen statisztika által meg tudjuk határozni az úgynevezett típus hibát, amire a gyártósoron oda lehet figyelni.

Végül tároljuk az operátor döntését az adott vizsgálatra vonatkozóan. A hibák száma és minősége alapján megfelel-e az elvárásoknak, vagy javítást szükséges végezni rajta, esetleg annyira elfogadhatatlan és javíthatatlan, hogy nem használható már fel semmilyen célra.

A hibákra vonatkozó táblák:

A hibák összetettek. Több csoportra vannak osztva, így a redundancia elkerülése érdekében több táblára bontottam ezeket az információkat.

Hibacsoport törzs:

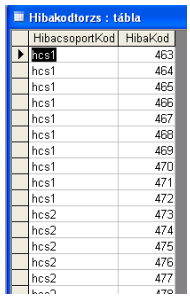
A leg alapvetőbb információja a hibáknak, hogy jellegük alapján csoportokra vannak bontva.

Azonosító	NyelvKod	HibacsoportKod	HibacsoportNev
1	ENG	hcs1	PTH_Insertion
2	HUN	hcs1	PTH_Beültetés
3	ENG	hcs2	PTH_Soldering
4	HUN	hcs2	PTH_Forrasztás
5	ENG	hcs3	SMT_Insertion
6	HUN	hcs3	SMT_Beültetés
7	ENG	hcs4	SMT_Soldering
8	HUN	hcs4	SMT_Forrasztás
9	ENG	hcs5	Manual_Insertion
10	HUN	hcs5	Kézi_beültetés
11	ENG	hcs6	Mech_Assembly
12	HUN	hcs6	Kézi_Összeszerelés
13	ENG	hcs7	Test
14	HUN	hcs7	Teszt
15	ENG	hcs8	Rework

Ez a tábla tartalmazza a csoportok megnevezését magyar és angol nyelven. A nyelvi bővítés során lehetőségünk van a táblát további adatokkal feltölteni. A kezelhetőség érdekében programozási szempontból bevezettem egy sorszámozott hcs sorozatot. Ehhez rendelem hozzá a hibacsoportok megnevezéseit nyelvenkén a nyelvi kóddal meghatározva.

Hibakód törzs:

A hibacsoportok és a hibák több nyelven történő megnevezése miatt készült egy olyan tábla, ami összekötő elem a konkrét hiba megnevezések és a hibacsoportok között. A többnyelvűség miatt a hibacsoportok „hibacsoport kóddal” lettek ellátva, a hiba megnevezések pedig sorszámot kaptak. Ez a tábla összegzi a közöttük lévő kapcsolatot, mely hibacsoportok milyen sorszámú hibamegnevezéseket foglalnak magukban.

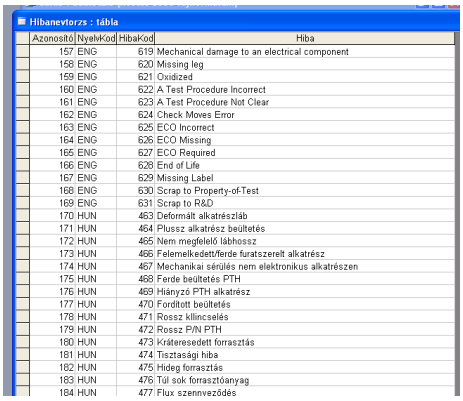


HibacsoportKod	HibaKod
hcs1	463
hcs1	464
hcs1	465
hcs1	466
hcs1	467
hcs1	468
hcs1	469
hcs1	470
hcs1	471
hcs1	472
hcs2	473
hcs2	474
hcs2	475
hcs2	476
hcs2	477

A gyakorlatban erre azért van szükség, hogy a vizsgálat során a felhasználó számára megjelenő hibalista ne több száz elemből álljon, melyek közül ki kell választani a konkrét hibát, hanem a hiba típusának meghatározásával leszűkítsük a hibák listáját.

Hibanév törzs:

Az NI központi hibakód-rendszerében szereplő konkrét hibamegnevezések tárolására szolgál ez a tábla. A szoftver többnyelvűsége miatt a tábla tartalmazza a nyelvi kódot, a hozzá tartozó idegen nyelvi megnevezést, és a hibakódot, amely beazonosítja, mely hibacsoporthoz tartozik az adott hiba.

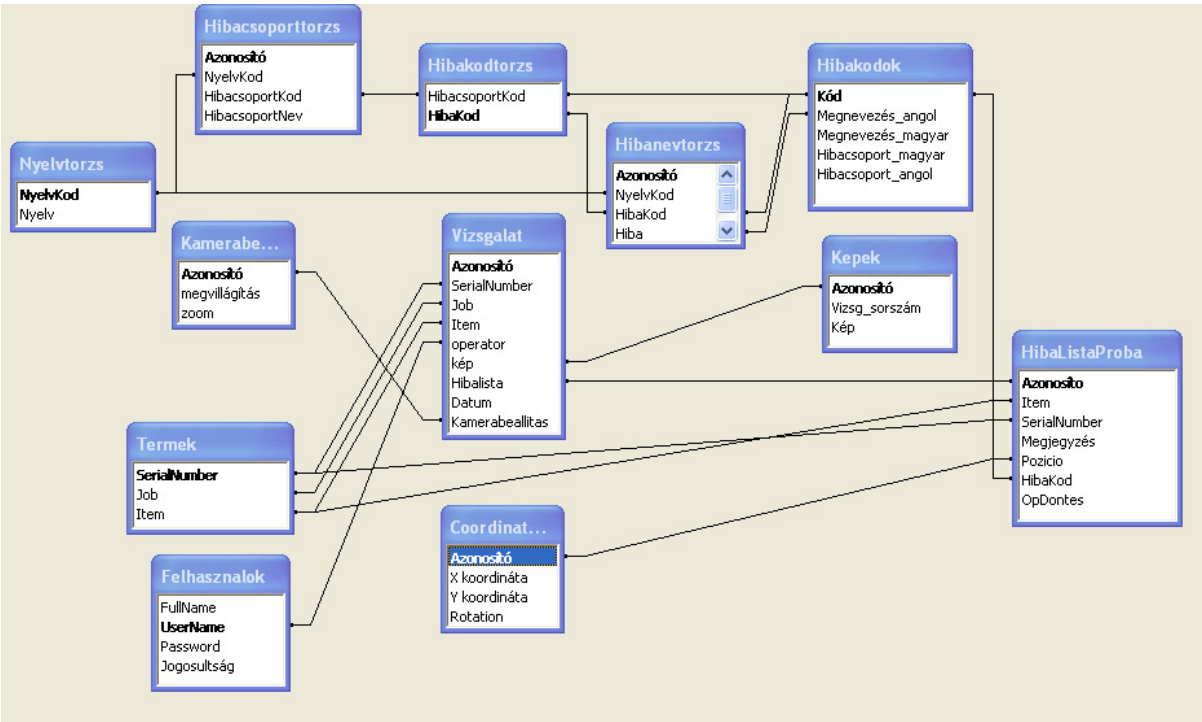


Azonosító	NyelvKod	HibaKod	Hiba
157	ENG	619	Mechanical damage to an electrical component
158	ENG	620	Missing leg
159	ENG	621	Oxidized
160	ENG	622	A Test Procedure Incorrect
161	ENG	623	A Test Procedure Not Clear
162	ENG	624	Check Moves Error
163	ENG	625	ECO Incorrect
164	ENG	626	ECO Missing
165	ENG	627	ECO Required
166	ENG	628	End of Life
167	ENG	629	Missing Label
168	ENG	630	Scrap to Property-of-Test
169	ENG	631	Scrap to R&D
170	HUN	463	Deformált alkatrészláb
171	HUN	464	Plussz alkatrész beültetés
172	HUN	465	Nem megfelelő lábhossz
173	HUN	466	Felemelkedett/ferde furatszerelt alkatrész
174	HUN	467	Mechanikai sérülés nem elektronikus alkatrészen
175	HUN	468	Ferde beültetés PTH
176	HUN	469	Hiányzó PTH alkatrész
177	HUN	470	Forditott beültetés
178	HUN	471	Rossz kilincselés
179	HUN	472	Rossz P/N PTH
180	HUN	473	Kráteresedett forrasztás
181	HUN	474	Tisztasági hiba
182	HUN	475	Hideg forrasztás
183	HUN	476	Túl sok forrasztóanyag
184	HUN	477	Flux szennveződés

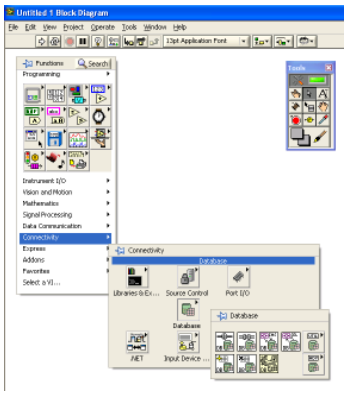
Az adattáblák közötti kapcsolat:

A szoftver adatbázistábláinak tartalmi információi széleskörűek. A program szempontjából a központi tábla a vizsgálatokról készült információkat tárolja el. Ehhez szükséges néhány már ismert adat, ezért kapcsolódik hozzá a felhasználóra vonatkozó paraméter és a kamera beállítások adatai. Az új információk tárolására két tábla szolgál, az egyik a vizsgálati panelről készült képekre vonatkozó információkat tárolja, a másik a képfelismerés során eredményül kapott eltérések listája.

A hibalista egyesével tartalmaz minden eltérést és annak tulajdonságait a pontos behatárolásra vonatkozóan. Ebbe a táblába kerül a hiba megnevezése is, ami a program nyelvi sokszínűsége miatt nyelvkódként kerül azonosításra. Ebből a kódból visszakereshető a hibacsoport, a felhasználói felületen történő megjelenítése csupán a nyelvi kód programindításnál kiválasztott értékétől függ.



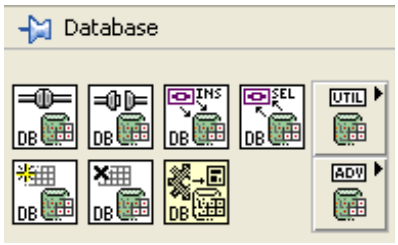
LabVIEW Database Connectivity Toolkit



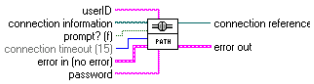
Az adatbázis kezeléshez külön fel kell telepíteni a LabVIEW-hoz a Database Connectivity készletet. [1]

A VPIS rendszerhez a 7.1-es verziót használtam, mely három csoportba sorolja a rendelkezésre álló beépített elemeket. Ezek a Database VIs, Utility VIs, Advanced VIs.

A Database VIs csoport elemei:

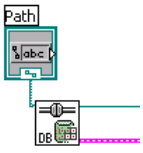


Open Connection



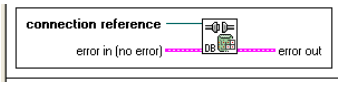
Az adatbázis megnyitására az Open Connection szolgál. Minden adatbáziskezelés ezzel az elemmel kezdődik, hiszen a táblákat csak megnyitott adatbázison keresztül tudjuk elérni.

Bemenő paramétere az a Connection information, amely megadja a használandó adatbázis elérési útvonalát. Kimenő paramétere a Connection reference, amely már megnyitott adatbázis és a hiba kimenet. Továbbá lehetnek egyéb bemenő paraméterei a felhasználóra, jelszóra vonatkozóan. A LabVIEW helpjére persze számíthatunk a bekötéseknél.

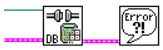


Close Connection- Kapcsolat bezárása

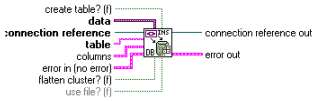
Az adatbázis bezárását végző művelet. A program használata végén végzendő el, vagy ha már nincs szükség az adatbázis adataira.



Bemenete a Connection reference és a hiba bemenet. Kimenete a hibakimenet.



Insert Data -Adat beszúrása

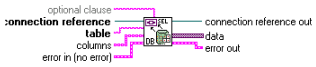


Az adatbázis tábláiban adatok elhelyezésére szolgáló művelet. Éppen ezért a bemente az az adat, amit a táblában el akarunk helyezni, mint elraktározandó információt. Meg kell adnunk a táblanevet, amelyik táblába el kell helyeznünk az adatot, és a táblán belül a mező nevét. A Connection Reference biztosítja az adatbázissal való kapcsolatot, hogy az adatbázis meg van nyitva, végezhetünk benne műveleteket. Továbbá van még egy hiba bemenet.

A művelet kimenete a Connection reference továbbadás és a hibakimenet.

Select Data - Adatok lekérdezése

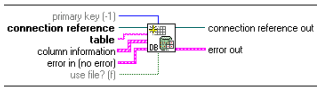
Egy konkrét tábla tartalmának lekérdezésre szolgáló elem.



Bemenete a Connection reference, hogy tudjuk, van kapcsolat az adatbázissal. Meg kell adnunk a lekérdezendő tábla nevét és azt, hogy a táblának mely oszlopára vagyunk kíváncsiak, továbbá rendelkezik egy hibabemenet is. A kimenet a Connection reference kimenete, a lekérdezés eredményeképp nyert adat és a hibakimenet.

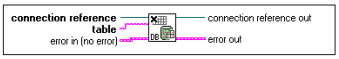
Create Table - Tábla létrehozása

Az adatbázisban új tábla létrehozására szolgáló művelet. Bemenete a connection reference, ami a kapcsolat létezését biztosítja. Meg kell adnunk a létrehozandó tábla nevét, az abban szereplő oszlopok nevét, adattípusát, méretét. Lehetőségünk van az elsődleges kulcs megadására is.



Az elem kimenete a Connection reference és a hibakimenet.

Drop Table-Tábla törlése



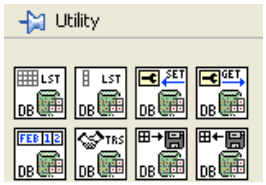
A művelet egy, az adatbázisban található táblát töröl teljes tartalommal együtt. Bemenete a Connecition reference, és a hibabemeneten kívül, a törlendő tábla neve. Az eszköz kimenete a Connection reference és a hibakimenet.

Variant To Data - Adattípus változtatás

Ez az elem átalakítja az adatbázisok típusait a LabView adattípusnak megfelelően, így lehetőségünk van az adatokat a VI egyéb funkcióira is használni.

Bemenete az adattípus, amivé át akarjuk alakítani az adatainkat, az adatbázisunkban található adatok, amiket át akarunk alakítani és a hiba bemenet. Kimenete a keletkezett adat és a hiba kimenet.

A Utility VIs csoport elemei:



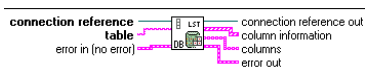
List Table-Táblák listázása

A megnyitott adatbázis tábláiról ad információt ez az elem. Segítségével megtudhatjuk, hogy az adatbázisunk milyen táblaneveket tartalmaz, a művelet kilistázza az adatbázis tábláinak a neveit.

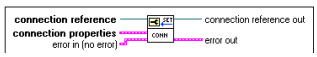
Bemenete a Connection reference és a hiba bemenet. Kimenete az adatbázis tábláinak a neve, a Connection reference és a hiba kimenet

List Columns-Oszlopok listázása

Egy meghatározott tábla oszlopait ismerhetjük meg ezzel a művelettel. Bemenete a Connection reference és a hibabemenet mellett a lekérdezendő tábla neve. A kimenet tartalmazza a lekérdezett tábla oszlopainak neveit és azok információit - név, adattípus és méret. További kimenete a szokásos connection reference és a hiba kimenet.



Set Properties-Tulajdonságok beállítása

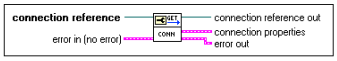


Beállítja az objektum tulajdonságait a bemeneti módon meghatározottan. Ez a VI kapcsolattal, parancsokkal, utasításokkal és paraméterekkel dolgozik. Bemeneteként a Connection reference, a hiba bemeneten kívül meg kell adni a Connection tulajdonságokaz. Ezek a tulajdonságok a parancs időhossz, a kapcsolat, és a kapcsolatellátó. Kimenete a Connection reference és a hiba kimenet.

Get Properties-Tulajdonság információk

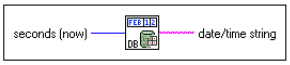
A bemeneti meghatározások alapján objektum tulajdonságokra vonatkozó információkat kaphatunk ennek az elemnek a segítségével.

Bemenő adatai a Connection reference, és a hibabemenet. Kimenetként szerepel a Connection reference és a hibakimenet mellett a kapcsolat tulajdonságokra vonatkozó lekérdezés eredménye.



Format Datetime Str – Aktuális időadatok sztringgé konvertálása

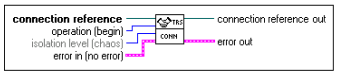
Az aktuális időpillanat értékeit sztringgé alakítja. Ez a művelet az aktuális idő adatbázisban való tárolására szolgál. A kimenő értéke fontos szerepet játszik.



Database Transaction – Adatbázis tranzakció

A tranzakció DML utasítások sorozata, a munka egyik logikai egységét alkotják. A tranzakció utasításainak a hatása együtt jelentkezik. A tranzakció sikeres végrehajtása esetén a visszagörgető szegmens információi átkerülnek az adatbázisba, véglegesítődnek. Ha valamilyen hiba folytán a tranzakció sikertelenül ér véget, akkor az visszagörgetődik és az adatbázis tranzakció előtti állapota nem változik meg.

Bemenő adatok a Connection reference és a hibabemenet mellett egy begin, commit vagy rollback művelet és egy elszigetelési szint, ami az egyidőben történő el nem döntött tranzakciókat kezeli.

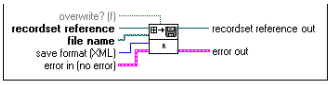


A művelet kimenete a connection reference és a hibakimenet.

Save Recordset To File – Rekordok fájlba mentése

Az adatbázis rekordjait lehetőségünk van XML vagy ADTG kiterjesztésű fájlba menteni. Ennek a mentésnek az elvégzésére szolgál ez a művelet.

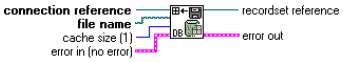
Bemenő paramétere a Recordset reference, amely az adatbázis elérhetőségére utal, a fájl neve, elérési útvonala -, amibe menteni szeretnénk a rekordokat. Meg kell adni a fájl formátumát, kiterjesztését is, ami XML, vagy ADTG lehet, végül a szokásos hibabemenet.



Kimenetei a Recordset reference és a hibakimenet.

Load Recordset From File – Rekordok betöltése fájlból

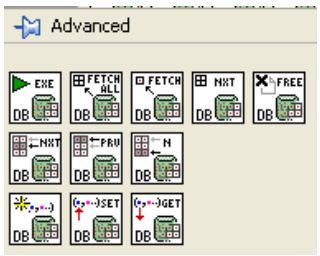
Az előző elem ellentéte, amely egy fájlból betölti a rekordokat és visszatér a recordset reference értékkel.



Benenő adatai a Connection reference, a betöltendő fájl neve, pontos elérési útvonallal és a hibabemenet. Megadhatunk egy számot is, hogy hány rekordot töltsön be az adott fájlból a memóriába. Ez az érték alapértelmezetten 1.

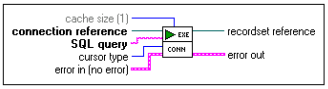
Kimenő paraméterei a Recordset reference és a hibakimenet.

Az Advanced VIs csoport elemei:



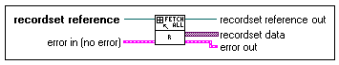
Execute Query -Lekérdezések

SQL lekérdezések végrehajtására szolgáló művelet. Bemenő paraméterei a connection reference, és a hibabemenet mellett egy SQL kifejezés és egy kurzor típus a rekordokra vonatkozóan. Kimenete a szokásos hibakimenet, és a recordset reference. [2,3]



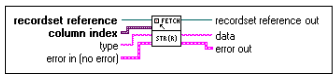
Fetch Recordset Data

Ez a művelet a Recordset reference azonos rekordjait válogatja le és egy két dimenziós tömbbe helyezi el az eredményt. Az elem bemenő paramétere a recordset reference és a hibabemenet. Kimenete a Recordset reference és a hibakimenet mellett a recordset adat. Az a tömb, amelyben a leválogatott adatokat a Variant to Data elem alkalmazásával konvertálhatunk vissza a LabView verzióról.



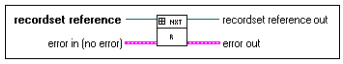
Fetch Element Data

Ez az elem egy adott táblán belüli elemkészlet beolvasására szolgál. Az azonos rekordkészletben az oszlopindexek aktuális feljegyzései határozzák meg az adatokat. Az oszlopindex lehet null-index pozíció az oszlopon belül – tehát egy szám - vagy az oszlop neve – mint szöveg. A bemenő típus „sokalakú” a visszatérési értéke pedig határozott típusú. Bemenő paramétere a recordset reference, az oszlopindex, a típus részletezve az oszlop adataira vonatkozóan, és a hibabemenet. Kimenete a recordset reference és a hibakimeneten kívül a keletkezett adat mező.

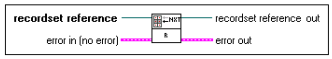


Fetch Next Recordset

Egy összetett rekordkészleten belül beolvassa a következő rekordkészletet. Az aktuális rekordkészlet tovább már nem vehető igénybe, a rekordset reference átirányítódik, az új rekordkészletre fog utalni. Bemenete a recordset reference és a hiba bemenet. Kimenete a recordset reference és a hibakimenet.

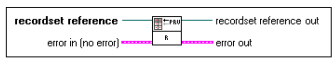


Move To Next Recordset



Ez a művelet egy lépést végez, hatására a kurzor a következő rekordra mutat, a rekordkészletet pedig a Recordset Reference azonosítja. Bemeneti paraméterei a recordset reference és a hiba bemenet. Kimenete a recordset reference és a hiba kimenet.

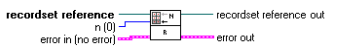
Move To Previous Record



Ennek az elemnek a hatására a kurzor visszafelé lép a rekordokon. Ez a VI nem használható egyirányú kurzorként. Bemeneti paraméterei a recordset reference és a hiba bemenet. Kimenete a recordset reference és a hiba kimenet.

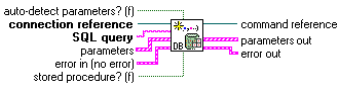
Move To Record N

Ez az elem a kurzort a rekordkészlet n-dik elemére lépteti a recordset reference által. A léptetés n és -1 közötti tartományban mozog és a kurzor nem egyirányú.



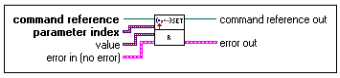
Bemeneti paraméterei a recordset reference és a hiba bemeneten kívül egy speciális számláló, amely n és -1 közötti értékeket vesz fel. Kimenete a recordset reference és a hiba kimenet.

Create Parameterized Query



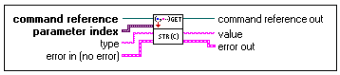
Adatbázis-műveletként készíthetünk paraméterezett SQL kifejezést. Bemenő paramétekként meg kell adni a connection reference mellett az SQL kifejezést, a hozzá tartozó paramétereket. A paraméter nevét, adattípusát, értékeit, és a paraméter értékére vonatkozó utasításokat. Megadható egy úgynevezett automata kimutatás paraméter, ami alapértelmezetten hamisra van állítva. Ha ez az érték igaz, akkor a LabView megkísérli automatikusan kimutatni a paramétereket az SQL kifejezésben, vagy tárolt eljárások nevében. Kimenő paraméterei a Command reference és a hibakimeneten kívül a paraméter kimenet.

Set Parameter Value



Beállítja a paraméter értékét a paraméter index alapján.A művelet bemenő paraméterei a command reference és a hiba bemenet mellett a paraméter index és az érték. A paraméter index egy speciális érték, vagy a paraméter index sorrendje vagy a paraméter neve. Kimenő paraméterei a Command reference és a hibakimenet.

Get Parameter Value



Egy paraméter index alapján meghatározza a művelet a paraméter értékét, amit valamely null-indexhez pozicionál. A bemeneti paraméter típusa sokoldalú, annak visszatérési értéke határozza meg az adat típusát. Bemente a Comman reference és a hibabemenet mellett a paraméter index és a paraméter típusa. Kimenő paraméterek a Command reference, a hiba kimenet és a paraméter visszatérési értéke.

Free Object

Felszabadítja az objektumot azáltal, hogy törli a hivatkozásokat. Bemeneti paramétere a Command reference, a hiba bemenet. Kimenetei a Connection fererence és a hibakimenet.

Adatbáziskezelés LabView DB Connectivity Toolki elemekkel

- Az elemek összekapcsolási módjai és lehetőségei -

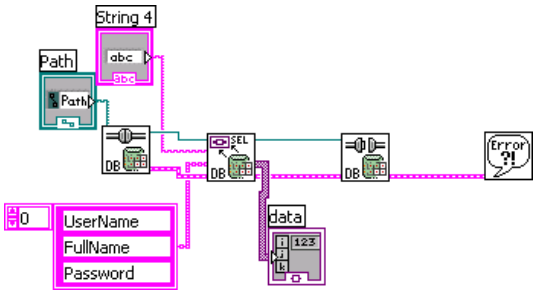
Adatbázis megnyitása és bezárása:

A legegyszerűbb műveletsorozat, ami tulajdonképpen semmi látványos dolgot nem tesz, mégis a legfontosabb tevékenység az adatbáziskezelés során.

A DB Connectivity elemei közül szükséges hozzá az Open Connection és a Close Connection.

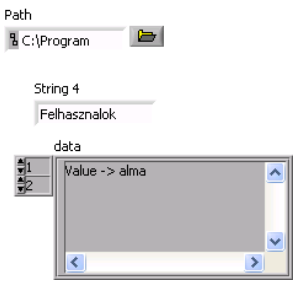
Az összekötéshez szükségünk van még két elemre. Az egyik az adatbázis elérési útvonalát meghatározó File Path Control. Ennek az elemnek a kimenete kapcsolódik az Open Connection elemhez, mint Connection information, és tovább halad a Close elembe. Az Open Connection kimeneteként megjelenik a hibakimenet, hiszen ha nem sikerül megnyitni az adatbázist, arról értesítést kell küldeni. Ez a hibakimenet belemegy a következő elembe, ami a Close Connection. Onnan egy Simple Error Handler-ben végződik, ami a felhasználó számára küld egy üzenetet a hibáról.

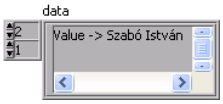
Egy adott tábla tartalmának lekérdezése:



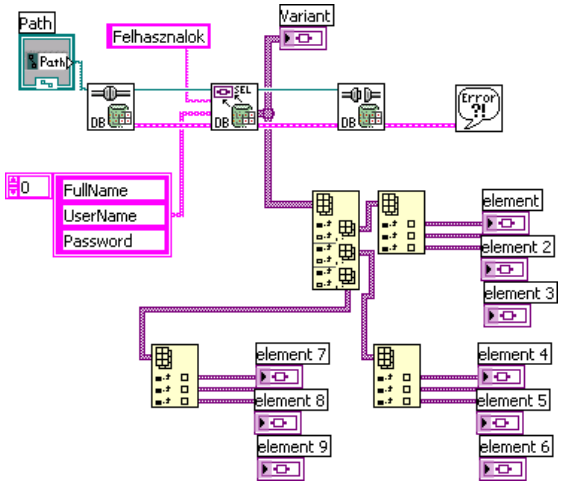
Következő lépésben annyival egészül ki az előző adatbázis megnyitó és bezáró kódrész, hogy a megnyitás és a bezárás elemek közé elhelyezünk egy *Select Data* műveleti elemet. Az Open két kimenetét a Select megfelelő bemeneteiként kötjük a folyamatba, kimenetei pedig a Close elem bemenetei lesznek.

A Select művelet elvégzéséhez meg kell határoznunk a lekérdezendő tábla nevét, amit egy String Control segítségével adunk meg a frontpanelen. Meghatározhatjuk azt is, hogy a tábla mely oszlopainak értékére vagyunk kíváncsiak. A művelet kimeneteként bekötünk egy kétdimenziós tömböt, amely tartalmazza a lekérdezés eredményét. Így a program futtatása során, ha a Felhasználók nevű táblát adjuk meg a sztringben és természetesen az adatbázis elérési útvonalát is pontosan határoztuk meg, akkor a lekérdezés eredménye megjeleni a tömbben:



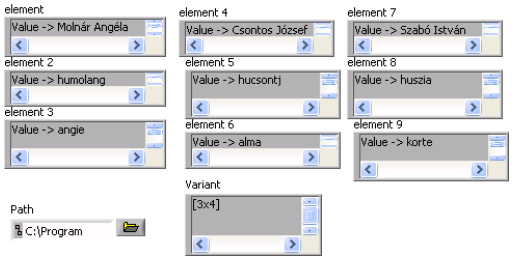


További próbálkozások a Felhasználók tábla lekérdezésére:

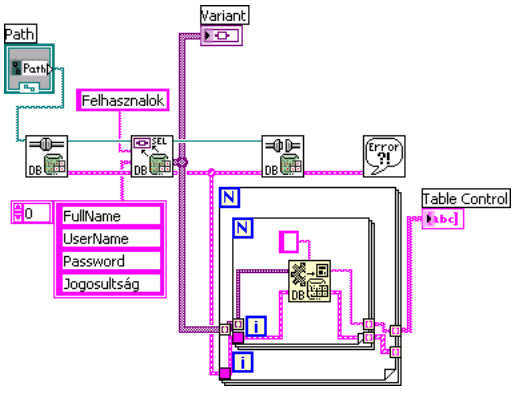


A Select elem data kimenete LabView specifikus. Erre a kimenetre egy Variant elemet kötve a program futása során a lekérdezés eredménye az eredménymátrix mérete, tehát [3x4].

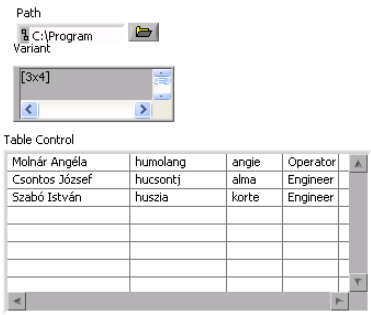
Ugyanerre a *data* kimenetre, ha tömbelemeket csatlakoztatok, dimenziónként szétszedve megjeleníthetem a lekérdezés minden eredményül kapott értékét. Az első tömbelem hatására rekordonként szedem szét az eredményt, majd azt külön-külön szétbontva, a rekord mezőit is megkapom. Ezáltal az *element*, *element2*, *element3* mezőkben helyezkednek el a lekérdezés első rekordjának mező értékei, az *element4-6*–ban a második rekord mezőértékek és az *element7-9*-ben a harmadik rekord tartalmai.



A lekérdezés ciklus segítségével, az eredmények megjelenítése táblában:

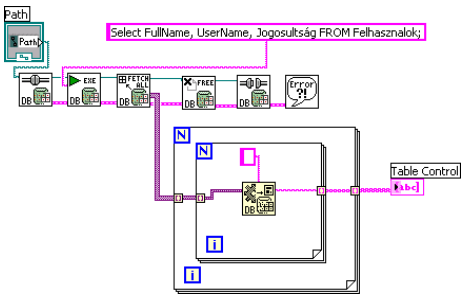


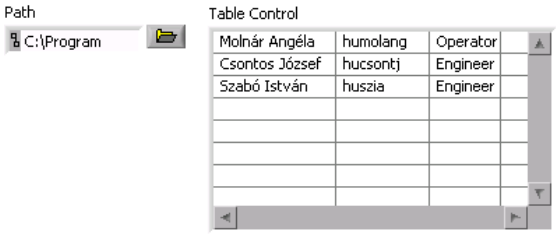
A lekérdezés során a tömbönkénti bontás művelete bonyolulttá teheti a programkódot, és hosszadalmas művelet a programíró szempontjából. Érdemes ezért egy ciklusba szervezni a tömbelemek szétbontását. Ezáltal a *Select* művelet *data* kimenet belemegy egy ciklusba, ami rekordonként járja végig az eredményt. Ezt követően a cikluson belül egy másik ciklust indítunk a rekordok mezőértékeinek megjelenítésére. A ciklusokon belül egy *Variant* művelet végzi a speciális LabView formátum átalakítását, az eredménye pedig egy *Table Control* elemben jelenik meg táblázatos formában. Ez a módszer egyszerűbb, biztonságosabb. Az eredmény megjelenése szemléletes és felhasználóbarát.



A lekérdezés végrehajtása speciális SQL kifejezéssel:

Az SQL kifejezéssel történő lekérdezést az *Execute Query* elem valósítja meg. Az előző programkódból kiszedtem a *Select* műveleti elemet, és helyére az *Execute* elemet kötöttem be. Ennek bemenő adata egy speciális SQL kifejezés, amely a Felhasznalok tábla három oszlopának értékét kérdezi le. Jelen esetben a kifejezés: *Select Fullname, UserName, Jogosultság FROM Felhasznalok;*.

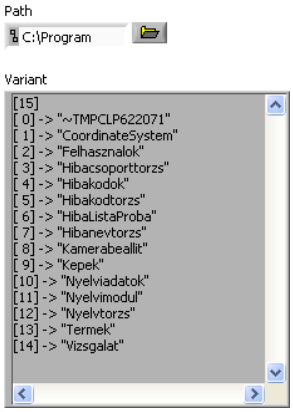


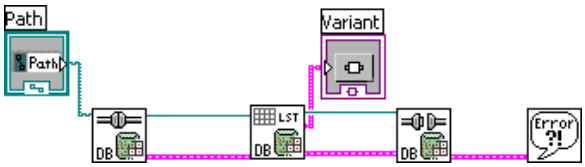


A műveleti elem kimenetét bele kell vezetni egy *Fetch Recordset Data* elembe, amelybe bekerülnek a leválogatott rekordok. Ennek *data* kimenetét a már jól megszerkesztett ciklusba vezetve a lekérdezés eredménye a Frontpanelen lévő táblázatban íródik ki. Érdemes egy *Free Object* műveletet bekötni a kapcsolatbezáró elem elé. Ez törli a lekérdezés utáni hivatkozásokat, memóriát szabadít fel.

Az adatbázis tábláinak lekérdezése:

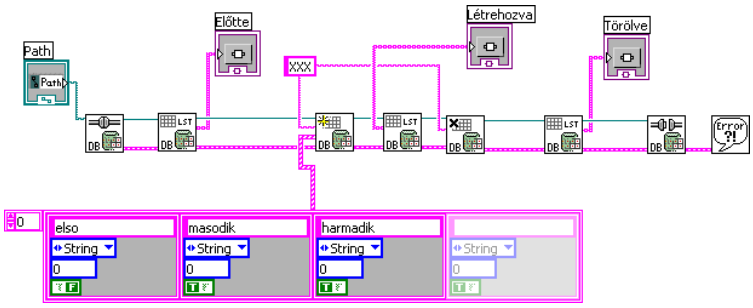
Az alapvető adatbázis megnyitó és bezáró programban egy *List Tables* VI-t helyeztem el. Az Open művelet kimeneteit a *List Tables* elem bemeneteiként kötöttem be, a kimeneteket a Close művelet bemeneteiként. A Frontpanelen elhelyeztem egy Variant elemet, amit a listázó művelet *data* kimenetéhez kötöttem. A program futásakor az elérési útvonalon található adatbázis megnyitásra kerül, a frontpanelen a Varianton belül kilistázódik az adatbázis tábláinak a neve soronként egymás alá helyezve, majd az adatbázis bezárásra kerül.





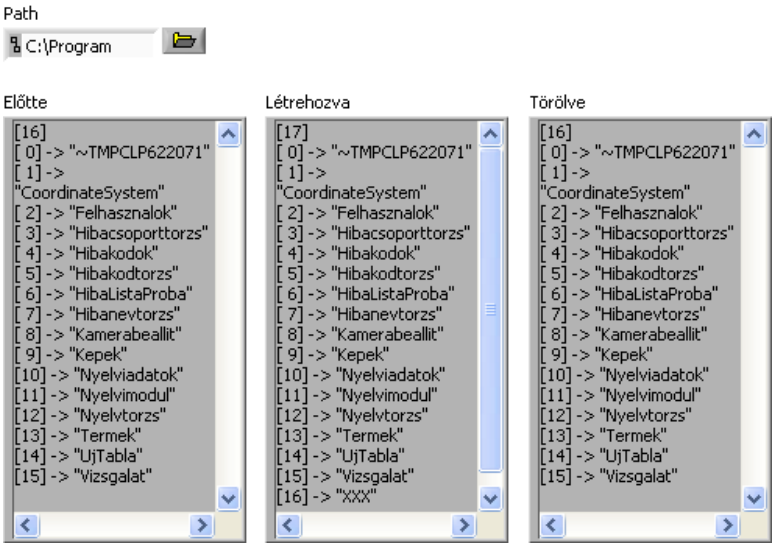
Új tábla létrehozása, létező tábla törlése:

Hogy láthassuk, valóban végrehajtódik a művelet, az új tábla létrehozása előtt is, és utána is lekérdezzük az adatbázis tábláit.



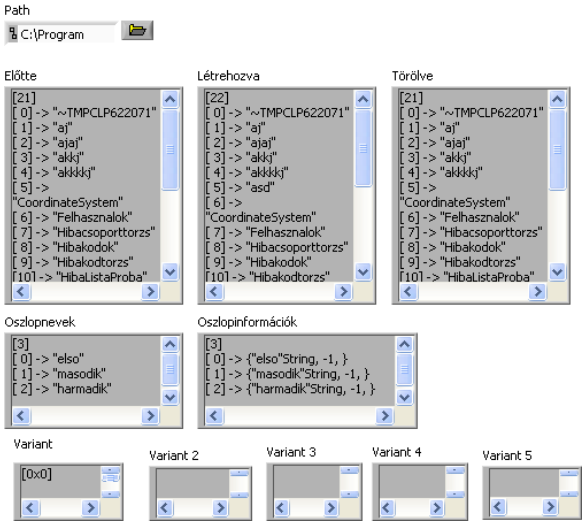
Először megnyitjuk az adatbázist, majd az előbb használt módon lekérdezzük az adatbázis tábláinak a neveit. Ez után egy *Create Table* művelet következik, amely létrehoz egy XXX nevű táblát a programozó által meghatározott oszloptulajdonságokkal. Hogy lássuk, valóban létrejött a tábla, a létrehozás után közvetlenül még egy *List Table* elemet helyezünk el. Annak érdekében, hogy a létrehozás műveletet többször is el tudjuk végezni, a lekérdezés után törlöm a létrehozott táblát. Ennek sikerességéről úgy tudok megbizonyosodni, ha egy harmadik táblanév listázó műveletet hajtok itt végre. Tehát a Drop Table elem Connection Reference out vezetéke a List Table Connection reference bemenetébe kötődik, a hibakimenet pedig hiba bemenetként, és majd csak ez után zárom be az adatbázist. A futtatás során a Frontpanelen nyomon tudom követni az elvégzett műveleteket. Első lekérdezés során 14 táblát tartalmaz az adatbázis. A létrehozás után a sor végére bekerült az XXX nevű tábla, majd a törlés utáni lekérdezés eredményében már nem szerepel csak az eredeti 14 tábla.

A lekérdezés során a táblanevek ábécésorrendbe rendezve jelennek meg.

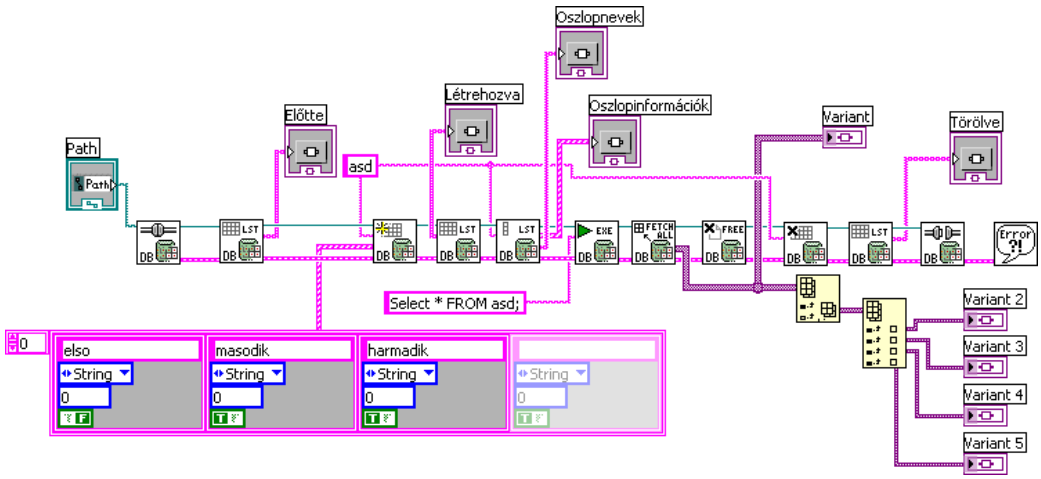


Összetett műveletvégzés:

Láthatjuk, hogy a LabView adatbázis-kezelése a Database Connectivity Toolkit elemek egymáshoz kapcsolásából áll. Legfontosabb tudnivaló, hogy az adatbázist meg kell nyitni a használata előtt, használata után pedig ajánlatos bezárni. Be nem zárt adatbázis esetén, ha újból futtatjuk a programot, hibaüzenetet kapuk arra vonatkozóan, hogy a program nem tudja megnyitni az adatbázist, mert az már eleve nyitva van. Ugyanilyen hibaüzenetet kaphatunk, ha már létező táblát szeretnénk létrehozni, vagy az adatbázisban nem létező táblát akarunk törölni.



A következő program megnyitja az adatbázist, lekérdezi az adatbázisban létező táblák neveit és kiírja őket az *Előtte* variantba. Ezután létrehoz egy *asd* nevű táblát az adatbázisunkban a megadott oszloptulajdonságokkal. Ennek szemléltetésére még egyszer lekérdezzük az adatbázis tábláit. Ezt követően a létrehozott táblára végzünk listázást. Kiíratjuk a tábla oszlopainak nevét és az oszlopok tulajdonságait. Ezek az *Oszlopnevek* és az *Oszlopinformációk* elnevezésű variantokban olvashatóak. Egy SQL lekérdezés hatására megtudjuk, hogy a tábla még nem tartalmaz rekordokat, mérete [0x0]. Ezt a táblát a programunkban ideiglenesen hoztuk létre, a továbbiakban nincs rá szükségünk, ezért töröljük a *Drop Table* művelettel. Ennek sikerességéről meggyőződhetünk egy újabb táblanevek lekérdezéssel, amelynek eredménye a *Törölve* nevű variantban jelenik meg. Végül bezárjuk az adatbázist.



A VPIS rendszerben használt adatbáziskezelések, adatbázis megvalósítások

A VPIS rendszer összetett feladatokon alapul. Szakdolgozati keretek között három nagy témára bontottuk, a kamerára vonatkozó ismeretekre, a vizsgálatot végző képfeldolgozásra és az eredmények eltárolása, statisztikai adatok nyerésére vonatkozóan az adatbázis-kezelésre. Az adatbázis műveletek végigkövetik a rendszer teljes folyamatát. A működés során felmerülő feladatokat modulokra osztottuk.

A VPIS rendszer fő moduljai és azok adatbázis vonatkozásai:

Adminisztrátor modul:

Célja a felhasználói jogosultságok kiosztása. Ezt a modult egy rögzített felhasználói névvel és jelszóval rendelkező adminisztrátor használja.

Az adminisztrátor modulban vizsgálandó az adminisztrátor bejelentkezése. Műveletei, tevékenységei az adatbázisra vonatkoznak. Elsődlegesen a felhasználókat és azok adatait rögzíti a rendszerben. Itt a Felhasznalok nevű táblába történik új rekord felvitele, illetve rekordok módosítása és törlése is előfordulhat.

Database modul:

Célja az adatbázis kezelése. Műveletei az adatbázis megnyitása, bezárása, adatok rögzítése táblákban, adatok lekérdezése táblákból, táblák közötti kapcsolatok megteremtése.

A szoftverhez tartozó *datab.mdb* adatbázisra és annak tábláira vonatkozó műveleteket hajt végre. A tevékenységet paraméter segítségével adhatjuk meg.

Fájlkezelő modul:

Célja a szoftver működése során a szükséges fájlok kezelése. A hagyományos fájlműveletek végrehajtása, mint a megnyitása, bezárása, mentés, mentés más néven.

Ennek a modulnak a jelenlegi adatbázis kapcsolata a vizsgálat során készített képekre vonatkozik. Az elkészült képek a számítógépen kerülnek tárolásra, az adatbázisban csak a rájuk történő hivatkozási cím tárolódik a *Kepek* táblában.

Image modul:

Célja a képjavítás. Előkészíti a képet a képfeldolgozáshoz képjavító műveletek végrehajtásának segítségével.

Ezek a képjavító műveletek előre meghatározottak, az adatbázis egyik táblájában találhatóak.

Kamera modul:

Célja az ideális beállítás meghatározása a képfeldolgozás számára készített kép minőségének javítása érdekében. A szoftver használata során alkalmazott képkészítő technikai berendezés kiválasztása utána a legideálisabb körülményeket határozza meg a képfeldolgozás szempontjából.

Az adatbázis *Kamerabeallit* táblája tartalmazza a mérnök által meghatározott megvilágítási és a zoom értékeket. A modulban a tábla tartalmából böngészhetünk, választhatunk.

Language modul:

Célja a felhasználó és a szoftver közötti nyelvi akadályok leküzdése a sikeresebb munkavégzés érdekében. A használat során a felhasználó igénye szerinti – angol vagy magyar – nyelven kommunikálhat a rendszerrel.

Ennek a modulnak az adatbázis igénye maximális. A szoftver elindítása során választandó ki a nyelvi specifikáció. Ez menet közben megváltoztatható, emiatt tökéletesen kell működnie. A rendszerben éppen ezért állandóan jelen van egy paraméter, ami a nyelvi kódot tartalmazza. Ennek megfelelően jelennek meg a formokon az elemek megnevezései, a hibaüzenetek, az opcionális értékek. A nyeli modul bővíthető, bármely nemzet nyelvére honosítható a szoftver.

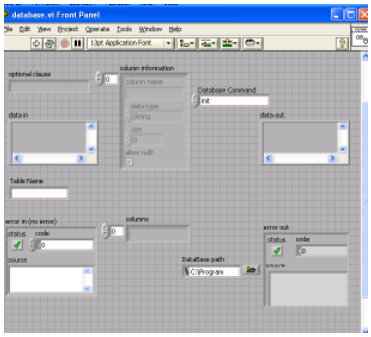
Login modul:

Célja a felhasználó beléptetése a rendszerbe. A szoftver használatának megkezdése előtt leellenőrzi, hogy az adott felhasználó jogosult-e a rendszerbe való belépésre. Ha jogosult, milyen feladatokat, műveleteket végezhet.

A modul adatbázis-műveletekre épül. A megadott felhasználói adatok alapján el kell tudni dönteni, hogy a *Felhasznalok* tábla rekordjai között szerepelnek-e az adott értékek, ha igen, milyen jogosultságokkal rendelkezik a bejelentkezett felhasználó.

Database modul

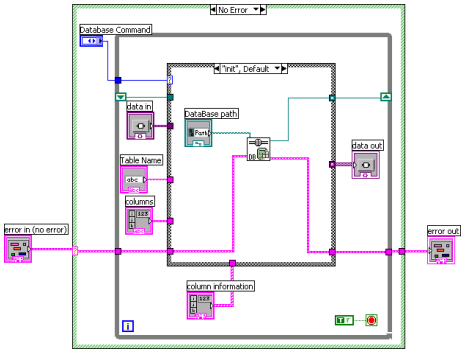
A database modul célja az adatbázis kezelése. Az adatbázis megnyitása, bezárása, adatok rögzítése táblákban, adatok lekérdezése táblákból, táblák közötti kapcsolatok megteremtése.



A modul bemenete a szoftver által használt *datab.mdb* adatbázis. A használat során paraméterként adjuk meg, hogy milyen műveletet akarunk végezni a modullal.

A modulnak megadható paraméterek:

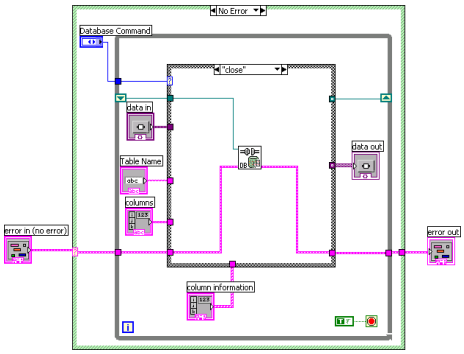
Inicializálás / init:



Az adatbázis megnyitására szolgáló művelet. A program indulásakor szükséges végrehajtani, hiszen már a bejelentkezés során a felhasználói adatok ellenőrzése is az adatbázisból történik.

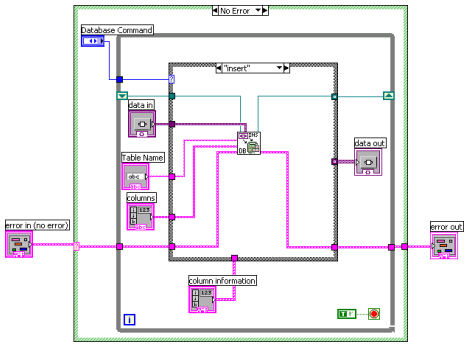
Az inicializásnál egyetlen bemenő adat a rendszer által használatos adatbázis, a *datab.mdb* pontos elérési útvonala. Az inicializás hatására megnyitjuk az adatbázist további műveletek számára.

Bezárás / close:



Az adatbázis bezárására szolgáló művelet. A program használatának befejezésekor szükséges végrehajtani, amikor már tovább nincs szükségünk arra, hogy az adatbázis elérhető legyen.

Beszúrás / insert:



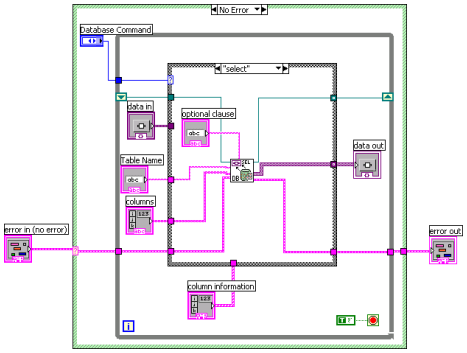
Adatok adatbázisban való eltárolására szolgáló művelet. A mérések során kapott eredményeket ennek a paraméternek a segítségével tárolhatjuk el az adatbázisban.

Bemenő adatként meg kell adnunk annak a táblának a nevét, amelyikbe szeretnénk az adatokat elmenteni, a tábla oszlopainak a nevét, és az elmentendő adatokat.

Egyszerű lekérdezés / select:

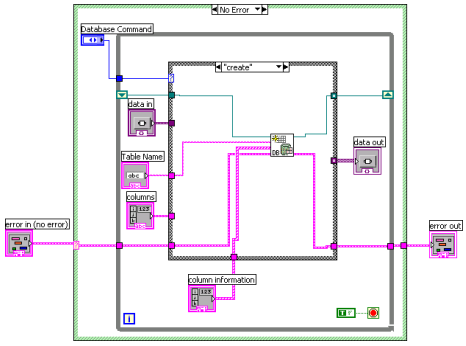
Az adatbázisban eltárolt értékeinkre kereshetünk rá ezzel a paraméterrel.

Lekérdezési feltételként megadhatjuk a tábla nevét, az adott tábla oszlopának nevét, amelynek értékeire kíváncsiak vagyunk, vagy speciális SQL kifejezéseket is szerkeszthetünk. Ebben az esetben az *Optional clause* értékének kell beírnunk az SQL utasítást.



A lekérdezés eredménye a *data out* kimeneten jelenik meg, ha nem történt hiba a lekérdezés során.

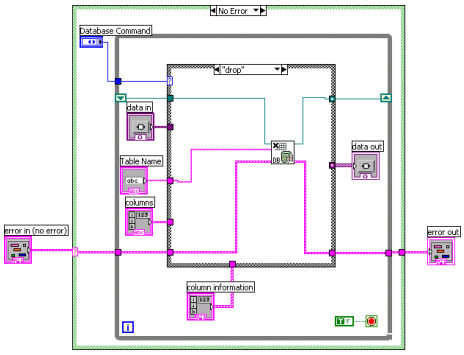
Tábla létrehozá / create:



A program működéséhez szükséges adatbázis rendelkezik táblákkal, amelyben a fontos és szükséges adatok szerepelnek. Működés közben a táblák létrehozására olyan esetben lehet szükség, ha ideiglenesen akarunk bizonyos adatokat eltárolni.

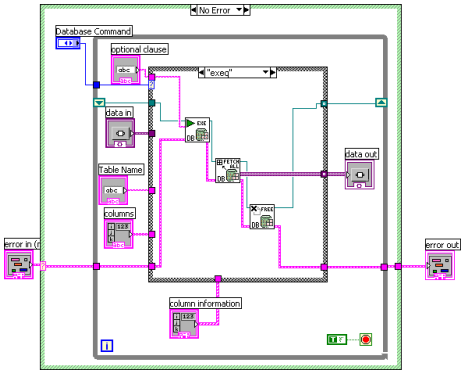
Meg kell adnunk a létrehozandó tábla nevét, és szerkezetét. Az oszlopok neveit, azok tulajdonságait.

Táblatörlés / drop:



Táblatörlésre a program során olyan esetben kerül sor, amikor ideiglenesen tárolunk adatokat az adatbázisban, arra ideiglenesen létrehozott táblában. Amikor tovább nincs szükségünk az ilyen táblákra, akkor ezzel a paraméterrel töröljük ki, a tábla nevének megadásával.

Lekérdezés / exeq:

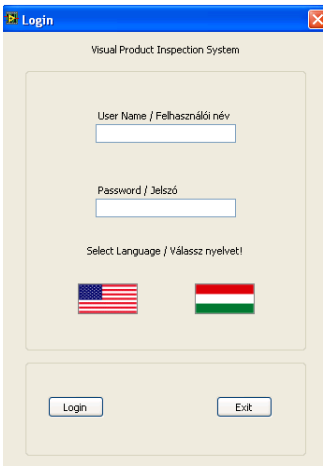


Összetett lekérdezések elvégzésére szolgáló paraméter. A szoftver működése során többször előfordul, hogy olyan összefüggéseket kell ábrázolni, amely nem csak egy adott tábla tartalmára vonatkozik. Ilyen esetekben speciális SQL kifejezéssel könnyen meg lehet szerkeszteni a szükséges lekérdezéseket.

Egy sztring formájában meg kell határoznunk az SQL kifejezést, ez lesz az egyetlen bemenő paraméter.

A szükséges execute, fetch és free elemek bele lettek építve ebbe a speciális adatbázis-kezelő VI-ba. A lekérdezés eredménye itt is a *data out* kimeneten jelenik meg, ha a lekérdezés minden gond nélkül, sikeresen végrehajtódik.

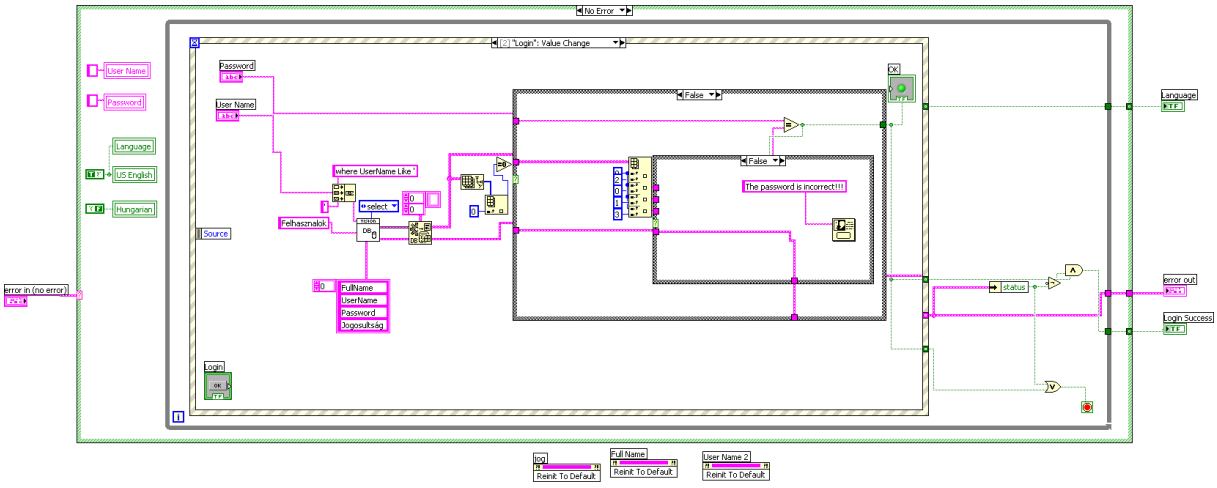
Login modul



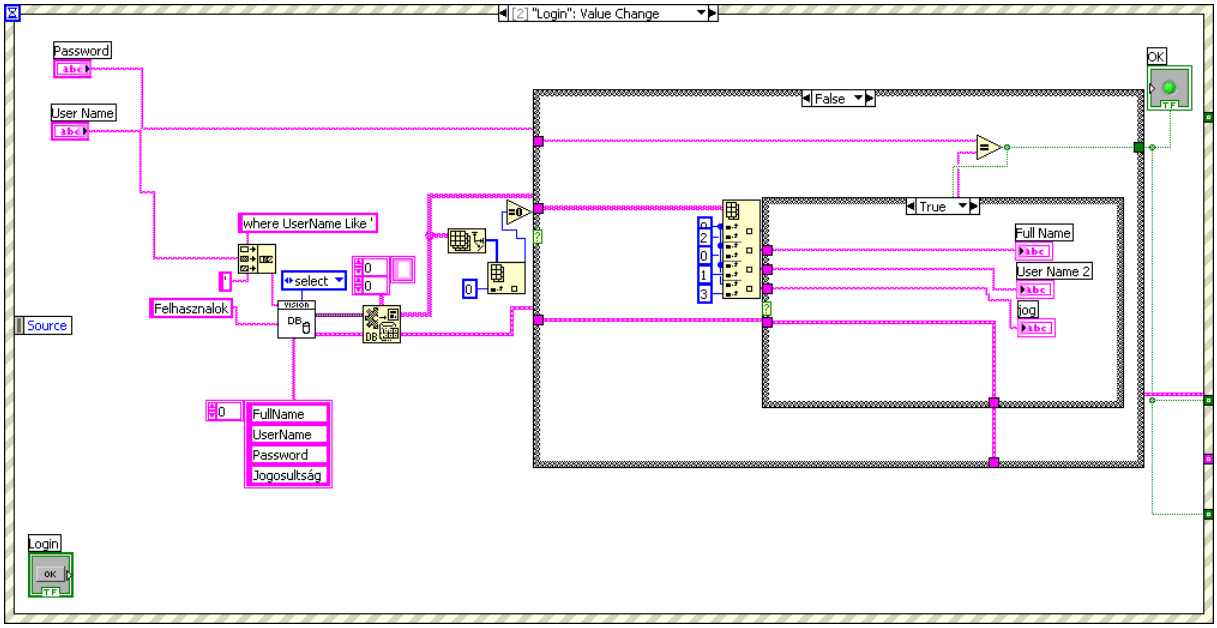
A Login modul célja a felhasználó beléptetése a rendszerbe.
A szoftver használatának megkezdése előtt leellenőrzi, hogy az adott felhasználó jogosult-e a rendszerbe való belépésre, ha jogosult, milyen feladatokat, műveleteket végezhet.

A modul bemenete a szoftver által használt *datab.mdb* adatbázis *Felhasznalok* nevű táblája, amely tartalmazza a felhasználói neveket, azokhoz tartozó jelszavakat kódolt formában és a felhasználók típusait jogosultsági körökre vonatkozóan.

A modul kimenete az adott felhasználó teljes neve, felhasználói neve, jogosultságára vonatkozó típusa, a rendszerbe lépésének pontos időpontja.



A login modul blokkdiagramjában láthatjuk a database modul felhasználását.
A modul párbeszédablaka kéri a felhasználói nevet és a jelszót. A database modul Select paraméterét használva a Felhasználók nevű táblában lekérdezést hajtunk végre a Felhasználói névre. Ha találtunk ilyen nevű felhasználót az adattáblában, akkor ellenőrizzük, hogy a megadott jelszó egyezik-e az adatbázisban szereplő jelszóval vagy sem.
A lekérdezés sikertelen eredményét közöljük a modul használójával, tudatva, hogy mi a gond, nem találtunk a beírttal azonos nevű felhasználót, vagy az adott felhasználói névhez tartozó jelszó nem jó.

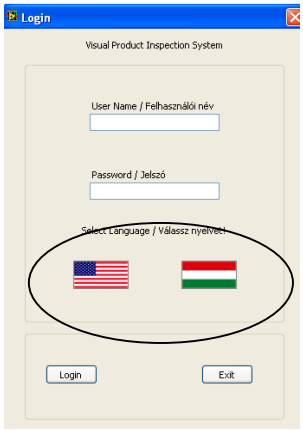


Sikeres bejelentkezés esetén kimenetre kerülnek a felhasználói adatok, a Teljes név, A Felhasználói név, és a felhasználó jogköre.

A jogosultság meghatározza, hogy az adott felhasználó a szoftver használata során milyen műveleteket hajthat végre. Bármely sikeresen bejelentkezett felhasználó végezhet vizsgálatot, de beállítási műveletek konfigurálására, statisztikai adatok lekérdezésére csak az Engineer szerepkör ad lehetősége.

Language modul

A Language modul célja, a felhasználó és a szoftver közötti nyelvi akadályok leküzdése, a sikeresebb munkavégzés érdekében. A használat során a felhasználó igénye szerinti (angol vagy magyar) nyelven kommunikálhat a rendszerrel.



A nyelv kiválasztására a Login modul ablaka ad lehetőséget. A felhasználó a bejelentkezésekor határozhatja meg, hogy milyen nyelven szeretné a rendszer feliratait és üzeneteit olvasni. Ha a bejelentkezés során nem él ezen lehetőségével, a cég nemzetiségét tekintve alapértelmezett az angol nyelv használata.

Az adatbázisban rögzítettek a panelen elhelyezkedő elemek feliratairа- és a hibaüzenetekre vonatkozó információk.

A nyelvi elemekre vonatkozó adatok adatbázisban tárolása lehetőséget ad a szoftver nyelvi bővítésére, ez által bármilyen létező nyelvre ki lehet terjeszteni a VPIS rendszer használatát.

Nyelvimodul : tábla

Azonosító	Vl	NyelvKod	Caption	Label
1	login	ENG	User Name / Felhasználói név	User Name
2	login	HUN	User Name / Felhasználói név	User Name
3	login	ENG	Password / Jelszó	Password
4	login	HUN	Password / Jelszó	Password
5	login	ENG		US English
6	login	HUN		US English
7	login	ENG		Hungarian
8	login	HUN		Hungarian
9	login	ENG		Login
10	login	ENG		Exit
11	login	HUN		Login
12	login	HUN		Exit
13	coordSys	ENG	Image	Image
14	coordSys	HUN	Kép	Image
15	coordSys	ENG	Coordinate System Out	Coordinate System Out
16	coordSys	HUN	Koordináta rendszer kimenet	Coordinate System Out
17	coordSys	ENG	Reference System	Reference System
18	coordSys	HUN		Reference System
19	coordSys	ENG	Measurement System	Measurement System
20	coordSys	HUN		Measurement System
21	coordSys	ENG	Origin	Origin
22	coordSys	HUN		Origin
23	coordSys	ENG	Angle (deg)	Angle (deg)
24	coordSys	HUN		Angle (deg)
25	coordSys	ENG	Axis Reference	Axis Reference
26	coordSys	HUN		Axis Reference
27	coordSys	ENG	Settings	Settings
28	coordSys	HUN		Settings
29	coordSys	ENG	Contrast	Contrast

VPIS DOKUMENTÁCIÓ, FELHASZNÁLÓI KÉZIKÖNYV

Áttekintés

Név:

Vizuális termékellenőrző rendszer – Visual Product Inspection System (VPIS)

I. Általános leírás:

A *Visual Product Inspection System* szoftver célja a National Instruments által gyártott egyes termékek vizuális ellenőrzésének automatizálása.

II. Általános követelmények:

1. A fejlesztők kötelesek minden üzleti logikához és felületi elemhez tartozó döntésükről kikérni a cég véleményét, egyetértését, hozzájárulását.

2. A rendszer felépítésének modulárisnak kell lennie, ami azt jelenti, hogy bármikor bővíthető kell, hogy legyen a cég fejlődési ütemének megfelelően, az informatikai rendszer minimálisnál nagyobb mértékű módosítása nélkül. Az újabb és újabb feladatokat ellátó modulok telepítésének egyszerűnek és gyorsnak kell lennie, más modulokat nem zavarhatnak meg működésükben.

3. Az alkalmazottak egész napos munkájukat gépnél töltik, ezért lényeges a felületek szép kinézete, a szemet nem bántó színek használata, a zavaró funkcióelrendezések kerülése. Fontos a következetesség, és a felület logikus kialakítása, vagyis az átláthatóság és könnyű kezelhetőség.

4. A rendszer minden művelet eredményéről tájékoztatja a szoftver használóját. Hiba esetén értesítést ad a probléma okáról a felhasználó minőségétől függően (például az operátor számára csak a munkáját nem zavaró mértékben kell utalni a probléma okára).

5. A rendszernek kezelnie kell a jogosultságokat is. Meg kell, hogy különböztesse az operátort a mérnöktől. Az operátorok csak az ellenőrzési funkciókat láthatják. A mérnököknek rendelkezésre kell, hogy álljon egy olyan bővített felület, mely lehetőséget biztosít a vizsgálat konfigurálására. Továbbá a szoftver biztosít egy adminisztrátori felületet is a rendszer adminisztrátora számára.

6. A rendszernek minden elvégzett és - sikeresen és sikertelenül – végrehajtott műveletről naplózást kell végeznie. Naplózni kell, az adatbázis és a szoftver működésének eseményeit, hibáit, stb. A naplózás szerkezetében el kell különülnie a különböző területekről érkező eseményeknek.

7. A rendszer statisztikai eredmények lekérdezését is lehetővé kell, hogy tegye a termékek vizsgálatára vonatkozóan.

8. A szoftvernek a cég által használt szakterületi fogalmakat kell használnia, melytől nem térhet el. Ennek részletes listázása a „Fogalomszótár” című dokumentumban található.

9. Forrasztott kötést nem kívánunk vizsgálni.

10. A rendszernek kezelnie kell a vonalkódokat.

11. A vizsgálandó termékek NYÁK méreteit figyelembe véve, a vizsgálati terület egy 20.5 x 15.7cm-es terület.

12. Az 1db kamera fix pozícióban áll és a tárgyasztalt sem kell mozgatni.

13. A rendszert a vizsgálatokhoz megfelelő, saját világítással kell ellátni.

14. A berendezés egy képből a legnagyobb biztonsággal végzi el a kívánt vizsgálatokat.

IV. Rendszer követelmények:

Operációs rendszer: Microsoft Windows XP

Programozási nyelv: National Instruments LabView 8.0

Célhardver: A szoftver igényeihez fognak igazodni. Szoftver tervezésekor nem kell figyelembe venni.

Várható felhasználók száma: 10 fő

Fogalomszótár

Admin/Adminisztrátor/Rendszeradminisztrátor

Olyan kitüntetett szereppel rendelkező felhasználó, aki a felhasználókat felveszi a rendszerbe, kezeli jogosultságaikat, jelszavaikat, vagy kitörölheti őket a rendszerből.

Engineer/Mérnök

Olyan felhasználó, aki a szoftver használata során minden joggal rendelkezik. Termékellenőrzést végezhet, statisztikai adatokat kérhet le a rendszertől, újabb ellenőrzésekre taníthatja be a szoftvert.

Felhasználók típusa

A szoftvert használó dolgozók csoportosítása különböző jogok alapján. Operator/Operátor csak termékellenőrzést végezhet, Engineer/Mérnök a termékellenőrzés mellett statisztikai adatokat kérhet le a rendszertől, vagy újabb műveleteket taníthat be neki.

Admin/Adminisztrátor/Rendszeradminisztrátor aki a felhasználókat a rendszerbe viszi, meghatározza típusaikat, kezeli jogosultságukat.

Item / Part Number

Termékkód, termék fajta, csoport azonosítója Pl.

Job

Egy termék gyártásának elindulása során a gyártandó termékek neve. Hozzá kapcsolódó adat, az Item szám/PN és az, hogy hány darab terméket gyártanak le belőle.

LabView

A National Instruments cég által kifejlesztett programozási nyelv és technológia, a termékellenőrző rendszer létrehozásához használt programnyelv.

NYÁK

Nyomtatott Áramköri Kártya, a National Instruments által gyártott termékek alapanyaga.

Operator/Operátor

Olyan felhasználó, aki a szoftver használata során csak termékellenőrző jogosultsággal rendelkezik, egy terméket ellenőrizni tud, de más műveletet nem végezhet.

Serial Number

Egy job gyártása során több azonos termék kerül ki a gyártósorról, ezek megkülönböztetésére szolgáló adat, egyedi azonosító. Ennek ismeretében be tudjuk azonosítani, hogy az adott termék mely jobhoz és mely itemhez tartoznak.

Statistic/Statisztika

A megfelelő jogokkal rendelkező felhasználó lekérdezheti a szoftver által már leellenőrzött termékek eredményeit. Az adott termékcsaládhoz tartozó kártyák milyen eredménnyel készülnek, milyen a hiba százalék, mik a jellemző hibák.

Visual Product Inspection System (VPIS)

Vizuális termékellenőrző rendszer

Forgatókönyv

1. Belépés a rendszerbe (Operator, Engineer)

- A belépni kívánó személy megadja a nevét és jelszavát.
- A „Belépés” gomb megnyomása.
- A rendszer ellenőrzi az érvényességet, és ezek alapján engedélyezi a belépést, vagy elutasítja azt. Ha engedélyezi a belépést, akkor eldönti a jogköröket is.

2. Vizsgálat

- JOB és ITEM szám megadása. Ha a felhasználó által megadott Job és Item szám nem létezik, akkor hibaüzenetet küld a rendszer, hogy az adott termékcsaládra nem tud vizsgálatot végrehajtani. Ha létező termékcsaládról van szó, a rendszer betölti a vizsgálathoz szükséges adatokat.
- Kamera beállítás. Ha az operátor behelyezte a vizsgálandó terméket, és kiválasztja ezt a funkciót, betöltődnek a szükséges paraméterek. A panelről elkészül egy kép, amit a szoftver leellenőriz, hogy használatra alkalmas-e.
- Vizsgálat. A rendszer képet készít a vizsgálandó panelről, majd összeveti azt az eredeti, helyes mintával. Elkészít egy hibalistát a két kép eltéréseiről. Vonalkód felismeréssel, karakter felismeréssel meghatározódik a Serial Number is.
- Hibalista végigjárása. A szoftver végigmegy sorban az által meghatározott hibalistán. Az operátornak itt felülbírálati joga van, eldöntheti, hogy a szoftver által talált eltérés valóban hiba-e.
- Serial Number ellenőrzése. A karakterfelismerő rendszer által meghatározott karaktersort a felhasználó vizuálisan leellenőrzi, szükség esetén korrigál, az adatok elmentése előtt.
- Vizsgálat befejezése. Ha egy termék VPIS által történt ellenőrzésével készen vagyunk, az adatokat tároljuk az adatbázisban, majd eldönthetjük, mit szeretnék tenni a továbbiakban, kilépni a rendszerből, új terméket ellenőrizni, esetleg felhasználót váltani.

3. Konfiguráció

- Adott termékcsaládhoz vonatkozóan elvégezhető kereső műveletek meghatározása.

4. Statisztika

- Adott vizsgálatok eredményeiről értesítést kaphatnak bizonyos felhasználók. Lekérdezhetők adott JOB-hoz tartozó vizsgált termékek száma, hibás termékek száma, a vizsgálatok során előforduló leggyakoribb hibák.

5. Kijelentkezés

- Ha a felhasználó befejezte a termékellenőrzést, akkor rákattint a „Kijelentkezés” gombra és a rendszer szabályosan kilépteti.

6. Kilépés a rendszerből

- Ha nincs már tovább szükség a vizuális termékellenőrző rendszerre (leállt a gyártás, vagy véget ért a munkaideje az ellenőrzést végző dolgozónak, vagy valamilyen ok miatt le kell állítani a számítógépet), akkor a „Kilépés” gombra rákattintva a rendszer leáll.

ÖSSZEFOGLALÁS

Egy termékellenőrző rendszer vizsgálatokat készít, a vizsgálatok eredményeit tárolja, statisztikát készít az adatok alapján. Ezen adatok tárolására elengedhetetlen a szakszerűen elkészített adatbázis, aminek megvalósítása az én feladatom volt a csapatmunka keretében.

A szoftver a National Instruments speciális LabView programozói környezetében íródott, emiatt meg kellett ismerkednem a programozási nyelv Database Connenctivity Toolkit eszközrendszerével és annak használatával. A szoftver működésének minden részén megjelennek az adatbázis-műveletek, emiatt rengeteg próbaprogramot készítettem gyakorlás céljából. Ezek egy részét megjelenítettem dolgozatomban segítségnyújtásként azok számára, akik adatbázis-kezeléssel szeretnének foglalkozni LabView környezeten belül.

A rendszer összetett, a vizsgálati elemek fokozatosan fejlődnek és bővülnek, emiatt a konkrétan elkészült modulok száma alacsony, de az adatbázismodul segítségével már bármilyen adatbázis-művelet egyszerűen beleépíthető a szoftver kódrészébe.

A VPIS adatbázisrendszere számtalan bővítési lehetőséggel rendelkezik. A nyelvi modul továbbfejlesztésével a szoftver más nyelveken is felhasználóbaráttá válhat, de az adatok tárolására használt jelenlegi Microsoft Access adatbázisrendszert is igény szerint lecserélhetjük például Oracle környezetre.

FELHASZNÁLT IRODALOM

[1] Database Connectivity Toolset User Manual a National Instrumentstől 2001. májusi kiadás

[2] Kishore Bhamidipati: SQL programozói referenciakönyv, Budapest: Panem Kiadó, 1999.

[3] Gábor András, Juhász István: PL/SQL-programozás, Budapest: Panem Kiadó, 2002.

[4] http://www.nieurope.com/

[5] http://ttk.unideb.hu/g-nap/felhivas.htm

[6] http://cnx.org/content/m13511/latest/

[7] http://office.microsoft.com/hu-hu/access/

KÖSZÖNETNYILVÁNÍTÁS

Ezúton szeretném megköszönni Dr. Szabó István egyetemi docensnek, hogy részese lehettem ennek a csapatmunkának, és szakdolgozatomat a Szilárdtest Fizika Tanszéken készíthettem el. Köszönöm, hogy munkámat figyelemmel követte, segítette, irányt mutatott a fejlesztések során.

Köszönöm Veres Péternek és Gyulai Tündének, a National Instruments Europe KFT mérnökeinek a segítségét, és a lehetőséget, hogy a gyár területén is dolgozhattunk, ezáltal bekapcsolódhattunk a termelés menetébe, figyelemmel követhettük a gyártási technológia lépéseit.

Továbbá köszönöm társaimnak, Cserép Imrének és Csontos Józsefnek, hogy együtt dolgozhattam velük, köszönöm az együtt töltött közös töprengéseket és megoldások keresése során született ötleteket.